Software Engineering Institute

Standards-Based Automated Remediation: A Remediation Manager Reference Implementation

Prepared for the
National Security Agency Computer Network Defense Research & Technology Program Management Office

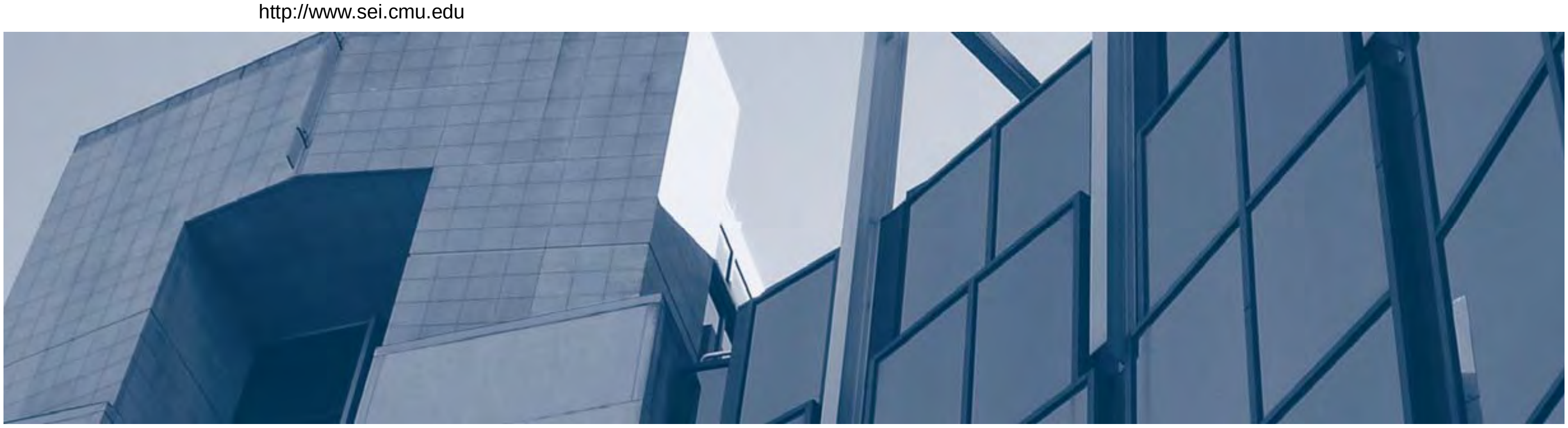
Sagar Chaki, Software Engineering Institute
Rita Creel, Software Engineering Institute
Jeff Davenport, Software Engineering Institute
Mike Kinney, National Security Agency
Benjamin McCormick, Software Engineering Institute
Mary Popeck, Software Engineering Institute

July 2011

SPECIAL REPORT
CMU/SEI-2011-SR-007

Acquisition Support Program; CERT® Program; and Research, Technology, and System Solutions Program

http://www.sei.cmu.edu



Carnegie Mellon

This material is based upon work supported by the United States Department of Defense under Contract No. FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

This report was prepared for the

SEI Administrative Agent
ESC/XPK
5 Eglin Street
Hanscom AFB, MA 01731-2100

For information about SEI publications, please visit the library on the SEI website (www.sei.cmu.edu/library).

Table of Contents

List of Figures

List of Tables

Acknowledgements

The authors would like to acknowledge the following members of the Software Engineering Institute's CERT® Program, who participated in the project by providing engineering and domain knowledge and support and participating in meetings and teleconferences: Rex Brinker, Chad Dougherty, Allen Householder, Chris Inacio, Drew Kompanek, Marty Lindner, and Art Manion. We also appreciate Tamara English's administrative support and Paul Ruggiero's editorial support. We thank Joe Wolfkiel of the Defense Information Systems Agency (DISA) for his guidance early in the project and for his continuing interest in our work. Finally, we extend appreciation to MITRE collaborators Matthew "Woj" Wojcik and Gerry McGuire and SPAWAR Systems Center Atlantic collaborators Jack Vander Pol, Kyle Stone, and Richard Kelly. Successful completion of this work would not have been possible without their contributions.

Abstract

This report describes the Software Engineering Institute's work in calendar year 2010 for the National Security Agency Computer Network Defense Research and Technology Program Management Office to develop standards for remediation of vulnerabilities and compliance issues on Department of Defense (DoD) networked systems. The overall goals are to assist in the development of remediation standards, demonstrate the functionality DoD would like in a remediation manager, and increase efficiency and effectiveness of remediation by automating the remediation process.

The 2010 Remediation Manager reference implementation demonstrates the following potential applications of remediation and other security automation standards: (1) Ingest scan findings in Security Content Automation Protocol (SCAP) format, extracting host compliance issues (in Common Configuration Enumeration [CCE] format) and vulnerabilities (in Common Vulnerability Enumerations [CVE] format). (2) Map CCE and CVE to remediation actions (in Common Remediation Enumeration [CRE] format). (3) Build remediation tasks in Remediation Tasking Language (RTL), based on CRE. (4) Transmit remediation tasks to a Remediation Tool on a host system. (5) Receive remediation task execution status, in RTL Results Format, from the Remediation Tool. This report identifies capabilities considered for future versions of the reference implementation and the operational system as well as challenges for future work.

1 Introduction

1.1 Remediation Research Overview

Existing methods and tools for remediating vulnerabilities and misconfigurations of Department of Defense (DoD) networked systems either rely heavily on manual support, which is inefficient and error prone and complicates delivery of remediation status data, or rely on proprietary vendor solutions.[1] The Remediation Research Project seeks to address these problems by (1) developing remediation standards, (2) increasing the efficiency and effectiveness of remediation by automating a remediation process that ensures host configurations comply with DoD policy, and (3) standardizing remediation processing.

The Remediation Research Project consists of four elements of work that advance efforts to develop standards-based, automated remediation capabilities:

- *remediation automation standards* (MITRE, NIST, Software Engineering Institute [SEI], SPAWAR Systems Center Atlantic, National Security Agency [NSA])
- *sample content*—security-related checklists, enumerations, and other information created in accordance with existing Security Content Automation Profile (SCAP) standards as well as the emerging remediation automation standards we are working to develop and test (G2, MITRE, NSA, SPAWAR Systems Center Atlantic)
- a *Remediation Manager reference implementation*—the subject of this special report (SEI)
- an *SCAP-based compliance checker* and *Remediation Tool reference implementation* (SPAWAR Systems Center Atlantic)

The remediation automation standards component of this work is based on the Derived Requirements (DR) identified by Waltermire, Johnson, Kerr, Wojcik, and Wunder [Waltermire 2011], which are shown in Table 1.

1 A *vulnerability* is a state in a system that allows an attacker to execute unauthorized commands, bypass restrictions on data access or modification, pose as another entity, or affect the availability of a system resource. A *misconfiguration* is any configuration state that does not comply with an organization's security policy. A *remediation* is a security-related set of actions that result in a change to a computer's configuration that brings it into compliance with policy (e.g., to address a vulnerability or misconfiguration) [Waltermire 2011, p. 1].

Table 1: Derived Requirements for Remediation Standards [Waltermire 2011]

ID #	Derived Requirement
DR1	method for uniquely identifying a remediation
DR2	definition of an exchange format for basic remediation information
DR3	definition of desired additional data about a remediation, including mappings to applicable platforms, related vulnerabilities, or configuration issues
DR4	definition of an expression language for the additional data about remediations as identified in DR3
DR5	method for specifying which remediations apply to which classes of assets
DR6	method for applying specific remediations to specific assets in an enterprise environment
DR7	method for reporting the results of an attempted remediation
DR8	method for expressing how to perform a remediation in a precise, machine-readable fashion *[Note: DR 8 is not part of the work described in this report and was rejected as a pursuit due to projected cost, complexity, concerns regarding the likelihood of success, and lack of vendor support.]*

Sample content, for use by the reference implementations, has been created as work on the standards progresses. Both the remediation standards and sample content are works in progress and should not be considered final.

The Remediation Manager reference implementation[2] ingests scan results, in DoD Assessment Results Format (ARF) version 0.41, which contain findings from host scans in the form of Common Configuration Enumeration (CCE) and Common Vulnerabilities and Exposures (CVE) entries. The Remediation Manager reads the policy for the given host's policy group. This policy maps CVEs and CCEs to a corresponding Common Remediation Enumeration (CRE) entry. Using the CRE, the Remediation Manager builds remediation tasks and transmits these tasks in Remediation Tasking Language (RTL) to the Remediation Tool associated with the host machine that requires remediation.[3] The Remediation Tool returns results to the Remediation Manager in Remediation Results Format (RRF). The Remediation Manager maintains a log indicating remediation task status (in process, failed, accomplished, not applicable, or undefined).

Figure 1 illustrates the role of the emerging standards in automated remediation management.

[2] The purpose of the reference implementation is to support development of remediation standards. The 2010 version does not incorporate all essential capabilities and quality attributes and is not a basis for operational system development.

[3] The 2010 Remediation Manager reference implementation accommodates scan results in DoD ARF version 0.41. Future versions will also accommodate scan results in Assessment Summary Results (ASR), eXtensible Configuration Checklist Description Format (XCCDF), and Open Vulnerability and Assessment Language (OVAL).

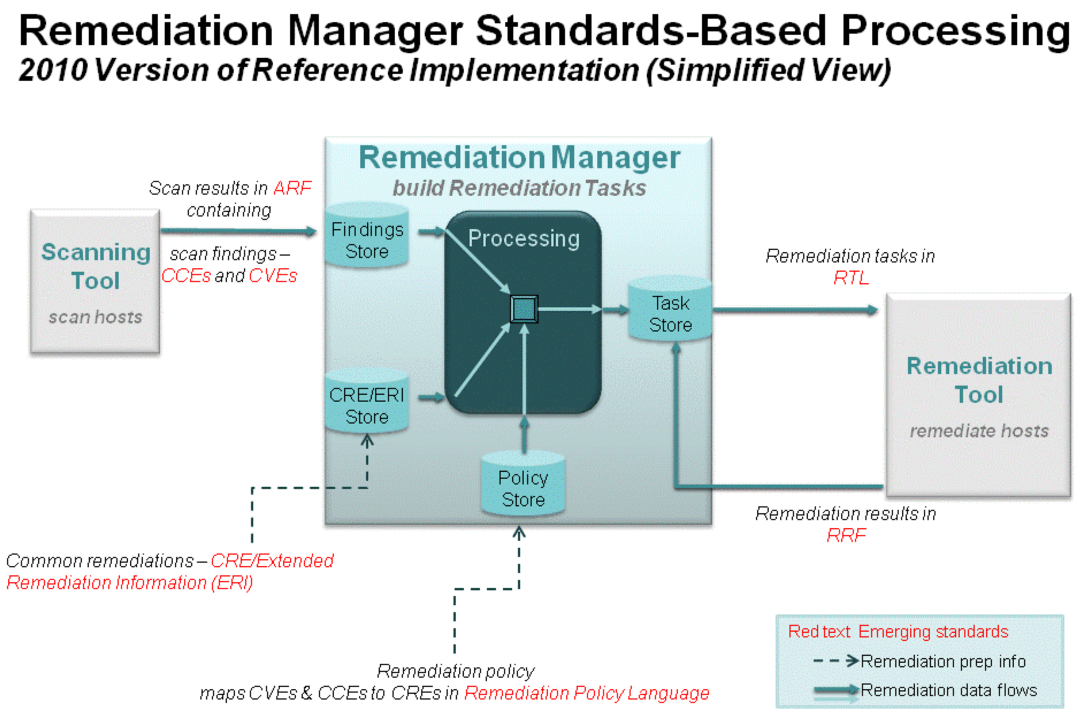


Figure 1: Standards-Based Processing for Automated Remediation Management

The Remediation Tool reference implementation consists of software that resides on a host system. This software receives remediation tasks from the Remediation Manager, executes these tasks on the host system, and sends task execution status back to the Remediation Manager. Initially, the Remediation Tool will be limited to remediating registry keys, file permissions, and local policy changes.

Note that in the 2010 implementation, the Remediation Manager assumes that the mapping from each scan finding (CCE or CVE) to each CRE has been defined in DoD remediation policy, and there is no need for user intervention. Future systems will include the ability for users to select from multiple CREs when necessary, to override DoD policy with local policy for machines that belong to certain policy groups, to choose a mitigation action, or not apply a remediation or mitigation at all. The user will also have the capability to enter justifications and build a Plan of Action and Milestones (POA&M) to handle deviations from DoD policy and output the results using SCAP version 1.1 standards.

1.2 Purpose and Organization of Report

The purpose of this report is to document the work accomplished on the Remediation Manager reference implementation in calendar year (CY) 2010 and to provide a technical foundation—including requirements, architecture, and design—for future work. In addition to describing the 2010 implementation, the report includes information on a broader set of requirements and on findings and observations to consider in defining the way ahead.

The report is organized into the following sections:

1. Introduction
2. Remediation Manager Vision and Scope
3. Remediation Manager 2010 Reference Implementation Development Project
4. Remediation Manager Requirements, Current and Future
5. Remediation Manager 2010 Reference Implementation Design
6. Remediation Manager 2010 Implementation and Verification
7. Project Challenges, Observations, and Next Steps

2 Remediation Manager Vision and Scope

2.1 Overview

This section describes the vision for the desired remediation management solution. It presents ideas developed during NSA's envisioning phase, describes the current context and the vision for the future, identifies key remediation management features, and illustrates the conceptual solution structure. Some of the goals for the Remediation Manager are implemented in the 2010 reference implementation, and others will be achieved through continued development in 2011.

2.2 Remediation Management Context

The Remediation Manager development effort has been defined to fit within the notional DoD network configuration management hierarchy shown in Figure 2, which illustrates the objective of leveraging standard remediations and policies defined at the highest level. While the objective of such reuse is laudable, lower-level tiers must retain the ability to tailor remediations and policies to address their respective mission objectives and risks.

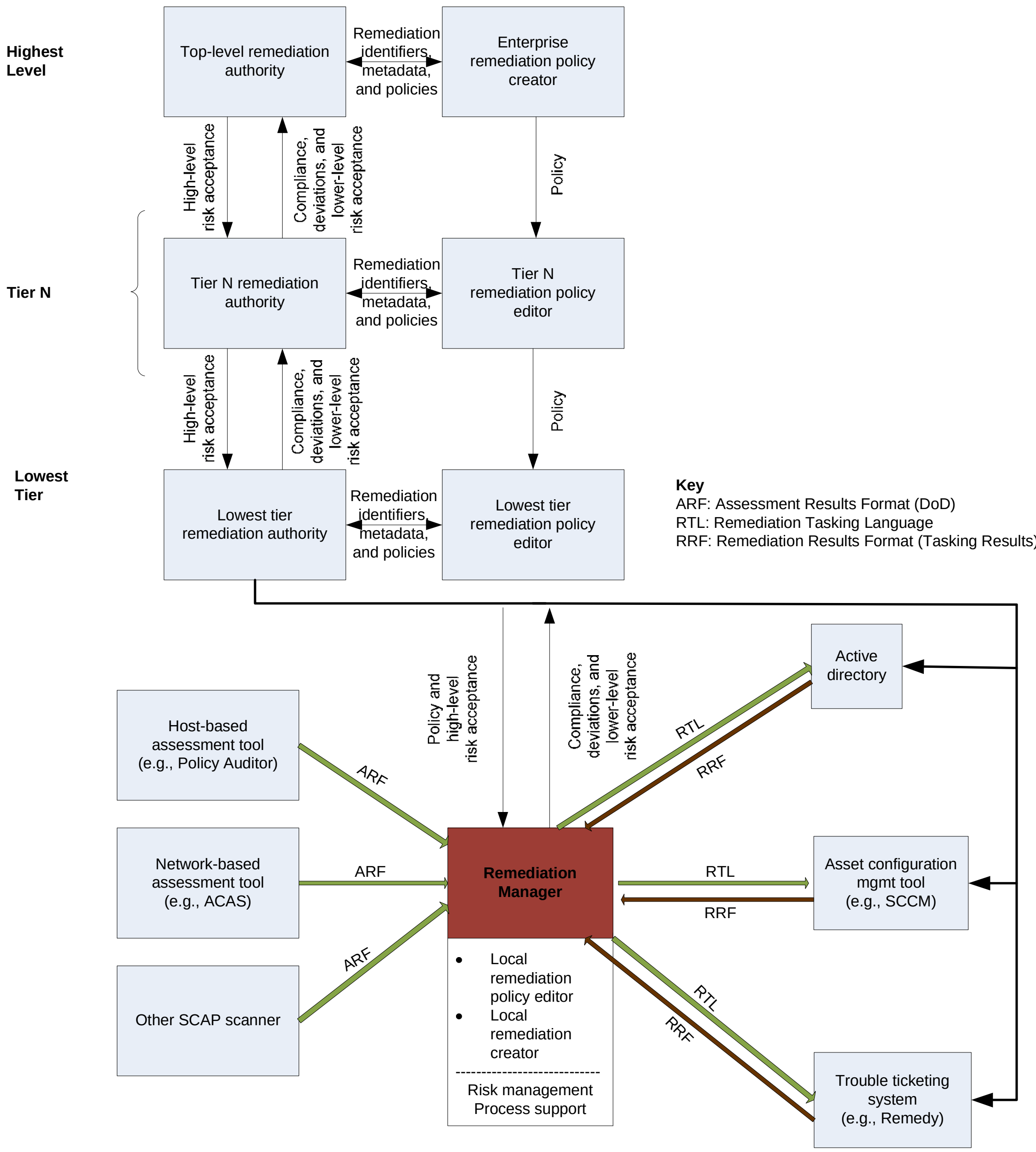


Figure 2: DoD Configuration Management Process Vision, Adapted from DoD[4]

At the bottom tier shown in Figure 2, assessment tools scan hosts (perform assessments) and produce results in standard formats. The use of standard formats for assessment data—which can be loaded into a repository, aggregated, correlated, deconflicted, interpreted, and processed—enables the following capabilities:

- Users of the assessment tools can manually organize, visualize, and understand assessment data.

4 U.S. Department of Defense. *Operational Concept Summary*. DoD, undated.

- The Remediation Manager described in this document can ingest the assessment data, along with remediation policy instructions that map assessment findings to remediation directives, and automatically output a directive (task) to apply a remediation.

The focus of this document is on the capabilities, characteristics, and development of a reference implementation for the Remediation Manager. The Remediation Manager will implement automated delivery and execution of remediation directives (also called tasks) for systems on DoD networks.

The work accomplished in developing the Remediation Manager reference implementation is expected to facilitate the DoD's procurement of standards-based remediation solutions via

- vendor development of standards-based, off-the-shelf components for various elements of the remediation solution
- an acquisition approach for an operational Remediation Manager implementation that is evolvable in capability and scale and meets specified functional, performance, and quality attribute (i.e., supportability and dependability) requirements

2.3 Significance of Standards-Based Automated Remediation

The DoD relies heavily on networked assets to perform its missions. These assets, and their interconnections, continue to grow in number and complexity. Maintaining a secure configuration—ensuring critical patches, settings, and updates are applied—is an ongoing challenge. The SCAP suite of security standards provides a means to express information about the configuration of networked assets and the results of scans so that prompt remediation and mitigation actions can be implemented. Emerging remediation standards will likewise provide an approach to expressing, selecting, and applying remediations to assets that are out of compliance or vulnerable to attack.

For remediation and mitigation to be prompt, automation is essential. The goal is to implement a standards-based, automated remediation solution that can be deployed within the DoD on enterprise-wide or isolated network enclaves (e.g., a tactical environment) to ensure that vulnerabilities and issues of noncompliance with DoD policy and guidance are corrected as soon as possible.

The vision for advancing vulnerability and configuration policy compliance will be realized in an evolutionary fashion and is described in Table 2 in terms of

- the current process (manual, supported by scripting and local methods and tools)
- basic capabilities to be provided by the reference implementation (2010 and 2011)
- capabilities under consideration for an initial operational capability to be acquired
- the desired final operational vision

Note that Table 2 represents the current understanding of desired capabilities. This understanding will evolve as work on the reference implementation and standards continues, resulting in changes to the planned capabilities for operational implementations.

Table 2: Remediation Manager Evolutionary Vision

Remediation Process Requirements (Current Process)	**Current Process: Manual Approach to Remediation** • Local information assurance (IA) users perform compliance scans and identify items to be remediated. • Scan results provided to local administrators who remediate manually, using scripting, local tools, and other methods. • IA users rescan and obtain Designated Approval Authority (DAA) acceptance for discrepancies. • IA users report results up the chain of command.
Refined Requirements (Reference Implementation)	**Reference Implementation: Research and Development of Automated, Standards-Based Remediation Manager** • Demonstrate how current scanning and remediation processes can be integrated and automated using SCAP and emerging remediation standards. • Perform automated compliance remediation actions based on preapproval of remediations (2010). • Automate remediation reporting, include POA&M(s) and statements of risk (2011).
Refined Requirements & Architecture (Initial Operational Implementation)	**Initial Operational Remediation Manager: Current Concept** • **Scope:** Limited scanning tool input and standards-based remediation policy govern limited patch and software setting configuration modifications. • **Remediation Policy:** The Remediation Manager ingests and stores CREs, ERI, (information associated with CREs), and remediation policy XML, which represents the baseline policy for all policy groups. Remediation policy maps CREs and required parameters to CVEs and CCEs. Using the local policy editor function of the RM, the local RM administrator may elect to apply local remediation policy rather than higher-level policy to a policy group of hosts. Overriding higher-level policy requires a justification/POA&M to document accepted risk and a time line (deadline) to bring the asset into compliance, which is reported in the remediation results. • **Host Assignment to Policy Group:** An administrator assigns each host in the Remediation Manager's inventory to the policy group that determines the set of remediation policies applied. The administrator may later decide to move a host into a different policy group. • **Scanning:** Scan results are sent to the Remediation Manager in the form of an Assessment Results Format (ARF) XML document. The Remediation Manager ingests the scan results and extracts findings (CCEs and CVEs). • **Policy Assignment for New Findings in a Policy Group:** The first time a finding is encountered for a host in a given policy group, the administrator is prompted to select default higher-level policy or local policy for that finding, for hosts in that policy group. If the finding requires different treatment for some hosts in a policy group, the administrator can move these hosts to a different policy group. • **Remediation for New Findings on a Host:** Once the remediation policy for a finding has been verified for a policy group, when a host in that group first encounters the finding, a remediation task will be sent to the appropriate Remediation Tool for execution on that host. The task status will be marked "in process" until the Remediation Tool returns a result status to the Remediation Manager. If remediation status is "failed," the machine is flagged and a ticket created so the host can be manually checked and remediated. • **Remediation for Repeat Findings on a Host:** If the host was previously scanned and tasked for remediation and the same finding is identified after the remediation deadline, the host is flagged and a ticket created requiring the host to be manually checked and remediated. • **Reporting:** Periodically, a report will be generated on all hosts in the Remediation Manager's inventory indicating findings from scan results and remediation status. This report will be an ASR report supplemented as needed to show remediation status information. The Remediation Manager administrator or user can manually generate a report at any time and display it at the Remediation Manager console. • **Remediation Manager Requests for Scans:** Newly discovered hosts will be placed into an "unassigned" group pending assignment to a policy group by the Remediation Manager administrator. When a new host is placed in a policy group, the Remediation Manager will prompt the administrator to request compliance scans or to request automated remediation without requiring a scan.
Refined Requirements & Architecture (Vision)	**Vision** • **Scope:** All devices on a network. • **Capability:** Data and logic to determine the best remediation option for a given host.

2.4 Vision Statement for Remediation Manager Reference Implementation

The Remediation Manager reference implementation should achieve the following objectives:

- Demonstrate the features documented in Section 2.5 of this document. (Remediation Manager increments 1 and 2, developed in 2010, demonstrate a subset of these features.)
- Support development of standards and associated content.
- Interface with others who are working on various aspects of security automation.
- Enhance understanding of the desired features of an operational Remediation Manager and remediation product suite implementation.
- Provide a technical foundation for development, procurement, or acquisition of an operational implementation via insights gained through reference implementation activities.

2.5 Remediation Manager System-Level Functional Requirements

Table 3 lists the top-level functional (feature) requirements defined for the Remediation Manager. Appendix A decomposes these requirements and allocates them to Remediation Manager component-level requirements and remediation standards requirements. It also identifies nonfunctional (quality) requirements. Note that not all system-level requirements have been implemented in the 2010 reference implementation, and some system-level requirements are only partially implemented. In some cases, this is because the necessary standards and content are not yet available; in others, it is because the functions were not allocated for implementation in 2010.

Table 3: System-Level Functional Requirements

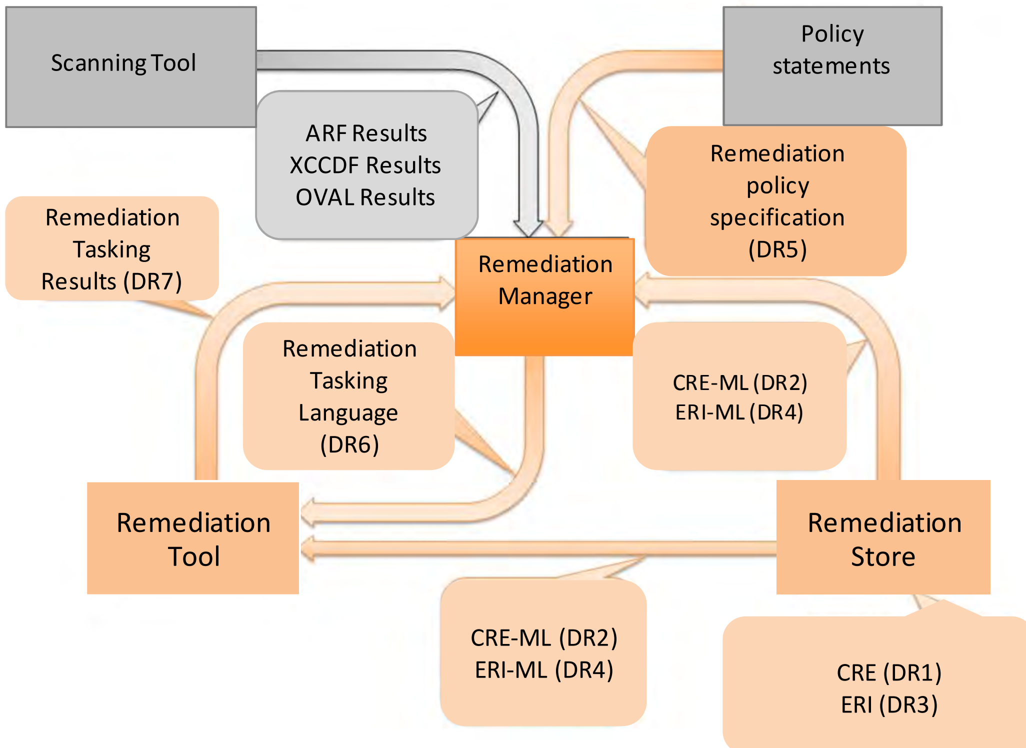
#	System-Level Requirement	Appendix A Reference
1	Accept input scan results formatted as • DoD’s ARF version 0.41 (implemented in 2010) • Assessment Summary Results (ASR), XCCDF, and OVAL (possible future)	Sys Remediation Manager 2.1 (ARF)
2	Accept input policy instructions consistent with standards-Derived Requirement DR5 [Waltermire 2011, NSA 2010[5]] (future).	Sys Remediation Manager 2.2
3	Output a directive to apply a remediation per standards-Derived Requirement DR6 [Waltermire 2011, NSA 2010[6]] (implemented in 2010).	Sys Remediation Manager 3.1
4	Allow users to choose which remediation to apply when multiple options are included in the policy (future).	Sys Remediation Manager 4.2
5	Determine the most efficient method of remediation (e.g., applying a single patch to fix multiple vulnerabilities) (possible future).	Not Applicable
6	Decide how to remediate when multiple remediation systems, including network-oriented systems, are available (possible future).	Not Applicable
7	Allow a user to tailor remediation policy for a given set of assets as well as accept some risks (i.e., decide not to remediate) (future).	Sys Remediation Manager 4.3
8	Assist users in building POA&Ms for policy deviations (future).	Sys Remediation Manager 4.4
9	Provide capability to publish POA&M messages consistent with Netops data standards (future). Note: For the reference implementation, when a deviation from policy is detected, the expected level of capability will be to reference a POA&M or make a mitigation statement (i.e., full POA&M capabilities are not a high priority for the reference implementation).	Sys Remediation Manager 3.5
10	Accept Remediation Tool results per standards-Derived Requirement DR7 [Waltermire 2011, NSA 2010[7]] (implemented in 2010).	Sys Remediation Manager 2.5
11	Republish findings received from the Remediation Tool with notations on fixes made (e.g., updating XCCDF results type to “fixed,” adding “info” messages to OVAL) (future).	Sys Remediation Manager 3.2

5 U.S. National Security Agency. *Integrated Statement of Work for FY2010 Remediation Concept Development.* NSA, 2010.

6 U.S. National Security Agency. *Integrated Statement of Work for FY2010 Remediation Concept Development.* NSA, 2010.

7 U.S. National Security Agency. *Integrated Statement of Work for FY2010 Remediation Concept Development.* NSA, 2010.

Figure 3 illustrates the DRs for remediation standards, identified and defined in Table 1, in the context of proposed remediation workflows. Note that not all these requirements will be incorporated in the reference implementation. Also, note that the terminology and design are continuing to evolve. For example, the Remediation Manager was formerly known as the Remediation Decision Tool, Remediation Tasking Language was called Remediation Control Language, and the role of the policy specifications has not been fully defined.



COLOR KEY		**ABBREVIATIONS**	
Gray	Existing components and data formats	ARF	Assessment Results Format (DoD v0.41)
Orange	Proposed data formats per Derived Requirements (DRs)	CRE	Common Remediation Enumeration
		DR	Derived Requirements
		ERI	Extended Remediation Enumeration
		ML	Metalanguage
		OVAL	Open Vulnerability Assessment Language
		XCCDF	eXtensible Configuration Checklist Description Format

Figure 3: Proposed Open Remediation Specifications (Derived Requirements) in the Context of Remediation Workflows [Wojcik 2009]

2.6 Remediation Manager Top-Level Functions

The Remediation Manager capabilities identified in Table 3 above can be grouped into three main functions:

1. Stage Policies (requirements 2, 4, 5, 6, 7, and 8)
2. Execute Policy-Based Remediations (requirements 1, 3, and 10)
3. Report Remediation and Risk Status (requirements 9 and 11)

The Stage Policies function obtains remediation policies, CREs, and Extended Remediation Information (ERI) from other systems and users and prepares them for use by the Remediation Manager. The Execute Policy-Based Remediations function extracts scan information, creates remediation tasks based on staged policies, transmits remediation tasks to Remediation Tool(s) or other mechanisms that will accomplish remediation, and receives remediation results. Finally, the Report Remediation and Risk Status function generates reports on the status of remediations and resultant risks, including risks and POA&Ms derived from policy exceptions.

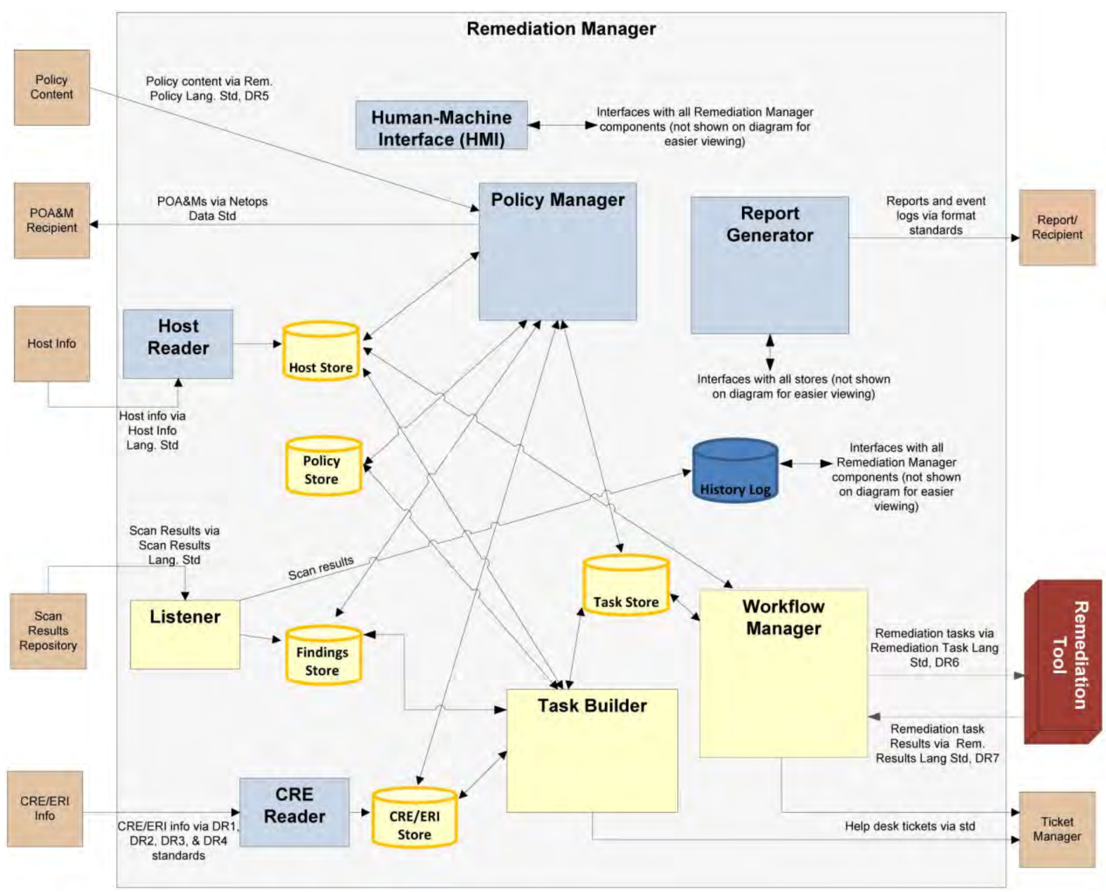
In 2010, reference implementation development focused on the second function, Execute Policy-Based Remediations. The next section briefly describes the project plan and schedule for developing the Execute Policy-Based Remediation capabilities of the Remediation Manager, which are implemented as three main Remediation Manager components: a Workflow Manager, a Task Builder, and a Listener.

3 Remediation Manager 2010 Reference Implementation Development Project

This section identifies the activities performed and the Remediation Manager reference implementation components created in CY 2010. These components were developed in two increments:

- Increment 1, delivered on September 15, includes the Workflow Manager, associated data stores, and the Remediation Manager interface to the Remediation Tool.
- Increment 2, delivered on December 10, includes Increment 1 plus the Task Builder, Listener, and associated data stores and interfaces.

Figure 4 illustrates a simple, conceptual architecture for the Remediation Manager. Yellow-shaded shapes indicate components that were developed in 2010. This is only a component view. Appendix A defines Remediation Manager requirements. Appendix B describes technical architecture views and the detailed design.



Note: Yellow-shaded components implemented in CY 2010.

Figure 4: Remediation Manager Conceptual Architecture

Table 4 identifies the activities performed to complete the CY 2010 work.

Table 4: Remediation Manager Development Activities

Activity Set 1 – Requirements Definition (baselined July–August with refinement as needed)
Document vision and scope
Develop requirements spreadsheet
Activity Set 2 – Top-Level Architecture and Design (baselined July–August with refinement as needed)
Develop conceptual architecture
Develop object model
Activity Set 3 – Workflow Manager Design and Implementation (Increment 1, delivered September 15)
Refine architecture and develop top-level design
Develop data tables
Design Workflow Manager and interface with Remediation Tool
Develop test scenarios
Implement Workflow Manager
Activity Set 4 – Listener & Task Builder Design and Implementation (Increment 2, delivered December 10)
Refine architecture and top-level design
Develop data tables
Design Listener and Task Builder
Develop test scenarios
Implement Task Builder
Integrate with Workflow Manager and other Increment 1 components
Activity Set 5 – Other Component Design and Implementation (concurrent with increments 1 and 2)
Develop log files
Integrate and deliver with Increment 1 and/or 2
Activity Set 6 – 2010 Findings and Final Report (delivered January 2011)
Document the following: • emerging understanding of requirements • basic architecture • trade-off decisions • lessons learned • questions (e.g., related to standards and balance of automation versus user control) • expected next steps

4 Remediation Manager Requirements, Current and Future

Appendix A documents Remediation Manager requirements implemented in 2010 as well as requirements envisioned for future increments. As such, it provides guidance for developing the remediation solution as a series of flexible, standards-based components rather than a single, monolithic package. Appendix A may be used as a basis for

- working on future reference implementations, including allocating capability requirements to deliveries, tracking implementation of requirements, and verifying and validating reference implementations
- tracking changes in requirements as research progresses
- clearly specifying the capabilities in the desired operational Remediation Manager solution

The appendix decomposes top-level (system) requirements into lower-level (subsystem) requirements and allocates these lower-level requirements to Remediation Manager subsystem components. System-level requirements deal with the interface between the Remediation Manager and the outside world, and subsystem-level requirements specify what each internal Remediation Manager component must do and how it interfaces with other internal Remediation Manager components and the outside world. System requirements may be partitioned among several subsystem components for implementation. These requirements, and their allocations to components, will most likely evolve.

The appendix is organized as follows:

A.1 Overview: description of requirements analysis and decomposition approach and requirements role classification

A.2 User Scenarios: five user scenarios defined for the Remediation Manager, mapped to the lower-level use cases that appear in the reference implementation design (Appendix B)

A.3 Remediation Manager High-Level Architecture: conceptual Remediation Manager high-level architecture, with subsystem components identified

A.4 Standards: identification of relevant SCAP specifications and standards

A.5 System Requirements with Decomposition to Remediation Manager Subsystem Components: tables identifying system-level requirements, associated Remediation Manager subsystem component requirements, requirement text, requirement source reference, and the increment to which the requirement was allocated for implementation

5 Remediation Manager 2010 Reference Implementation Design

Appendix B documents the Remediation Manager reference implementation architectural views, detailed design, and XML schema for the Remediation Manager-Remediation Tool interface as follows:

B.1 Introduction: purpose, scope, and acronym definitions

B.2 Architectural Views

- use case view: 10 use cases
- object view: class model, state diagrams, and flow chart
- component view: conceptual architecture diagram

B.3 Detailed Design

- four software modules
  - Administrator (part of the Task Builder subsystem component)
  - Listener
  - Processor (part of the Task Builder subsystem component)
  - Workflow Manager
- interfaces
- data store

B.4 XML Schemas

- interface protocol used to communicate remediation tasks (from the Remediation Manager to the Remediation Tool)
- remediation task results (from the Remediation Tool to the Remediation Manager)

6 Remediation Manager 2010 Implementation and Verification

6.1 Overview

The various Remediation Manager subsystem components were implemented in Java. The stores were implemented as tables in a MySQL database. The Remediation Manager subsystem components interact with the stores via Java Database Connectivity (JDBC). The Remediation Manager implementation is available in source and executable formats. It is available both as a stand-alone package and within a VMware virtual machine running RedHat Enterprise Linux 5. Both versions come with installation and execution instructions.

6.2 Packages

The Remediation Manager implementation consists of four main packages. Appendix B describes the Remediation Manager design in more detail.

1. *database*—This package consists of helper classes to interact with the store via JDBC. It consists of three subpackages:
   - *lightweight*—This package consists of lightweight objects, such as host, finding, and so on, for various elements of the data model.
   - *persistence*—This package consists of manager classes that mediate interaction between the Remediation Manager components and the stores.
   - *test*—This package consists of classes for unit testing each of the manager classes in the *persistence* package mentioned above.
2. *taskbldr*—This package contains the implementation of the Listener as well as the implementation of the Task Builder subsystem component and its subcomponents, the Administrator and the Processor.
3. *workflowmgr*—This package contains the implementation of the Workflow Manager subsystem component and a dummy remediation tool used for various tests.
4. *testgen*—This is a helper package for instantiating XML schemas.

Detailed documentation is available as part of the Remediation Manager release in Javadoc format.

6.3 Verification

The Remediation Manager was verified via both unit and integration testing. During unit testing, each manager class within the *database.persistence* package was exercised by using it to

1. create randomly generated entries in the store
2. modify these entries in a random manner
3. delete the entries

After each step, we checked the store to ensure that appropriate entries were present (or absent). The classes that perform these unit tests are in the *database.test* subpackage. This testing has since been supplemented by JUnit test classes that exercise each method for all of the *database.persistence* and *database.lightweight* classes.

During integration testing, the entire Remediation Manager was exercised, using the *taskbldr.test.FullTestHarness* class, to simulate the five use-case scenarios described in Table 5.

Table 5: Remediation Manager Verification Scenarios

Verification Scenario	**User Scenario #s** *see Appendix A.2*	**Use Case #s** *see Appendix B, Section B.2.1*
1. Scan results were received by the Listener, but the hosts involved in the findings were not assigned to any policy groups. The Task Builder created appropriate tasks for the Policy Manager (the Policy Manager subsystem component is not part of the 2010 version of the Remediation Manager).	1	B.2.1.3 B.2.1.6
2. Scan results were received by the Listener. The hosts involved in the findings were assigned to appropriate policy groups. The Task Builder created appropriate remediation tasks. The same scan results were read in for a second time by the Listener while the earlier remediation tasks were still “in process” and not past their due time. The Task Builder associated the second batch of findings with the corresponding earlier remediation tasks.	1	B.2.1.3 B.2.1.4 B.2.1.7
3. Scan results were received by the Listener. The hosts involved in the findings were assigned to appropriate policy groups. The Task Builder created appropriate remediation tasks. The same scan results were read in for a second time by the Listener after the earlier remediation tasks were past their due time. The Task Builder created new remedy tasks for the Ticket Manager and associated the second batch of findings with these remedy tasks (in the 2010 version of the Remediation Manager, there is no interface with an actual Ticket Manager).	1	B.2.1.3 B.2.1.4 B.2.1.7
4. Scan results were received by the Listener. The hosts involved in the findings were assigned to appropriate policy groups. The Task Builder created appropriate remediation tasks, and the Workflow Manager sent them to the Remediation Tool. The Remediation Tool returned “success” results for all tasks. The Workflow Manager updated the status of all remediation tasks to “accomplished.”	1, 2	B.2.1.3 B.2.1.4 B.2.1.5 B.2.1.9
5. Scan results were received by the Listener. The hosts involved in the findings were assigned to appropriate policy groups. The Task Builder created appropriate remediation tasks, and the Workflow Manager sent them to the Remediation Tool. The Remediation Tool returned “failure” results for all tasks. The Workflow Manager converted all the remediation tasks to remedy tasks and changed their target to the Ticket Manager.	1, 2	B.2.1.3 B.2.1.4 B.2.1.5 B.2.1.8

7 Project Challenges, Observations, and Next Steps

7.1 Development Challenges

The Remediation Manager development team faced three major challenges. The first was a learning curve with respect to the security domain, the SCAP standards, and the DoD Configuration Management Process. The second was the fact that SCAP remediation standards are still in early stages of development, so both the standards and the content they specify were evolving during Remediation Manager development. Finally, capability requirements for the Remediation Manager are continuing to evolve as well.

The reference implementation effort has advanced the development of remediation standards and matured the concept of automated remediation. It has placed some structure on the capability requirements development process while enabling requirements to continue to evolve. The next section describes key observations and questions identified during the Remediation Manager requirements and design processes.

7.2 Observations and Questions for Consideration

During the Remediation Manager development project, the team identified several questions for consideration as the project moves forward. These questions involve concepts of operations for end-to-end DoD remediation, both the ideal and what can be achieved in the near term. Key topics the team has identified as needing further exploration, discussion, experimentation, and articulation include

- the balance between automation and user intervention in the Remediation Manager and the Remediation Tool as well as the allocation of functions between these two elements of the end-to-end remediation solution
- hierarchical and peer-to-peer relationships with respect to reporting and other types of information sharing
- the extent to which standards-based remediation management can be centralized and coordinated across DoD, and different architectural strategies for accomplishing key coordination goals
- policy management, including automating the evaluation of new and updated policies to identify conflicts and compromises, keeping policies current and consistent, and adjudicating and reporting conflicts between global and local policies

These and other topics need to be shared and discussed in terms of the wider standards, DoD Configuration Management, and security solutions vendor communities.

7.3 Expected Next Steps

The Remediation Manager development team is planning several activities in 2011 to support continued evolution of an automated, standards-based remediation process. Among these are the following:

- Extend the 2010 reference implementation to provide additional capability, including a local policy editor and associated interfaces; deadlines, prioritization, and ticketing; the ability to interface with additional remediation tools and to store and process tool capability data; and results and status logging and reporting.
- Study and provide feedback on scalability of remediation management capabilities, deployment options (e.g., Host Based Security System [HBSS] and disk), and other topics relevant to an operational, standards-based remediation manager.
- Support the remediation standards development process through standards community participation.
- Drive the development or acquisition of operational remediation solutions by characterizing essential features and identifying technical challenges (e.g., developing candidate architecture descriptions and analyzing capability and quality attribute requirements).

In addition to the above tasks, the development team recommends work in the following areas:

- Extend Remediation Manager capabilities to address key challenges, for example, by developing a smart Policy Manager to analyze policies when updates or overrides occur and warn of possible inconsistencies, ambiguities, or attacks.
- Implement a measurement capability that could be used to support both enterprise and local decision making by
  - analyzing and reporting on remediation statistics and trends, for example, to identify host machines that most often fall out of compliance or need repeat remediation
  - determining which methods, practices, and tools provide the most (and least) benefit and value

7.4 Conclusion

Work on the Remediation Manager Reference implementation is proceeding in accordance with Remediation Research goals of advancing development of an automated, standards-based remediation approach for the DoD. In 2010, the team delivered initial increments of capability based on SCAP and emerging remediation standards and content. In 2011, development will continue with added remediation management capability, support to the standards development process, and identification and community discussion of key technical and management challenges.

This effort has been a model example of effective collaboration across a number of organizations, including NSA, MITRE, SPAWAR Systems Center Atlantic, and the SEI. The SEI team is committed to continuing and extending its collaboration with all those who share an interest in automated, standards-based remediation as we move this critical work forward, from research to operational capability.

Appendix A Remediation Manager Requirements

A.1 Overview

This section describes the system requirements analysis and decomposition to lower levels. The described requirements specify what has been built for Increments 1 and 2 of the prototype system,[8] as well as what assumptions were made, and what we envision will be built in future Remediation Manager increments. Note that in this document, the requirements are not comprehensive for the future vision. Future analysis and design efforts will likely result in modifications to the current set of requirements.

In general, requirements are statements of need that define what the Remediation Manager will do and how well it will do it. At lower levels of the system, the requirements include specifics on what each subsystem component must do, how the subsystem component will interface with other parts of the Remediation Manager, and what part the subsystem component will play in the overarching requirements. The highest level of requirements, the system level, defines the interface between the Remediation Manager and the outside world. Requirements at this level describe how the outside world will interact with the Remediation Manager and what the Remediation Manager will do in response. To obtain lower-level requirements, the Remediation Manager was conceptually broken up into a set of interacting subsystem components, and the system-level requirements were decomposed to those subsystem components. A system requirement may be implemented in whole by a particular subsystem component; however, the requirements are often partitioned between several subsystem components.

Requirements may also be classified by the role that they play in the system. We have chosen the following classifications:

- standards and external interfaces—requirements that specify how the Remediation Manager interfaces with the outside world and the limitations that it must stay within
- inputs—requirements that specify what inputs will be provided to the Remediation Manager and what the Remediation Manager is expected to do upon receipt
- outputs—requirements that specify what outputs the Remediation Manager is expected to provide
- user interface and functions—requirements that specify how a user or operator will interact with the Remediation Manager and what the Remediation Manager is expected to do in response
- Remediation Manager internal functions—requirements that specify the Remediation Manager capabilities and what the Remediation Manager must do
- Remediation Manager nonfunctional requirements—requirements that specify either how well the Remediation Manager must perform a function or exhibit the quality attributes (e.g., dependability, supportability, and usability)

8 Increment 1 was delivered in September 2010, and Increment 2 in December 2010.

Traceability for the requirements has also been documented. The system-level requirements are linked from their more general form (i.e., as specified in the request for information, statement of work, user scenario, or other source document) to their more concrete form (e.g., the subsystem components). Traceability allows us to verify that the appropriate subsystem component implements all higher-level requirements and that each subsystem component implements only approved requirements that can be traced to a user requirement.

A.2 User Scenarios

This section includes five user scenarios for the Remediation Manager. Only user scenarios 1 and 2 have been implemented in the Increment 2 deliverable. User scenarios 3, 4, and 5 are for a future delivery but illustrate the future Remediation Manager vision. These user scenarios map to the use cases, which are defined in Appendix B, Section B.2.1. Table 6 shows the user scenarios and their use case mappings.

Table 6: User Scenarios and Use Case Mappings

User Scenario Number	User Scenario Title	Related Use Cases as Defined in Appendix B, Section B.2.1
1	Perform a Remediation Based on Findings and Policy Content	B.2.1.3 Scan Hosts B.2.1.4 Handle Findings B.2.1.5 Handle New Findings B.2.1.6 Handle Unassigned Hosts B.2.1.7 Handle Repeat Findings
2	Receive and Save Remediation Status and Results	B.2.1.8 Process Failed Tasks B.2.1.9 Process Successful Tasks
3	Allow a User to Edit Policies, CRE/ERIs, Host Information, Remediation Tasks, Remediation Results, and Reports	
4	Allow a User to Assign Hosts and Policies to Policy Groups	B.2.1.1 Assign Policies to Policy Groups B.2.1.2 Assign Hosts to Policy Groups
5	Allow a User to Generate a Status Report	B.2.1.10 Print Remediation Report B.2.1.11 Auto-Generate Remediation Report

A.2.1 User Scenario 1: Perform a Remediation Based on Findings and Policy Content

On a periodic basis, the Remediation Manager receives policies, scan results, and host information, according to the specified standards, and stores them. Upon receipt of the scan results, the Remediation Manager extracts the noncompliant finding information (i.e., CCE or CVE issue), marks the finding as "new," and stores it.

The Remediation Manager reviews each noncompliant finding on each assessed host. If the host has not been assigned to a policy group, the Remediation Manager flags it and user scenario 4 is invoked.

For hosts that have been assigned to a policy group, the Remediation Manager determines if the host was previously scanned and if there were any open findings (i.e., CCE or CVE issues) against it. If the

host was previously scanned and if the same finding is still open, then the Remediation Manager determines if the remediation task that was generated to close the finding has passed its due time. If the due time has not passed, then the Remediation Manager just records the host's status. If the due time has passed, then the Remediation Manager updates the remediation task status to "failed" and outputs a help desk ticket for the host to be manually checked and remediated.

If the host has been assigned to a policy group and a new finding has been found, then the Remediation Manager maps the finding to a corresponding CRE and then generates a remediation task with a due time to close the finding. The Remediation Manager assigns the new remediation task to the appropriate Remediation Tool and transmits it to that tool per the specified standard. Once a remediation task is generated and sent, the Remediation Manager sets the task status to "in process." If the host does not require remediation, then the Remediation Manager simply records its status.

When the Remediation Manager finds a noncompliant finding that has more than one potential remediation action that could be performed to remediate the host, then the Remediation Manager outputs a help desk ticket.

A.2.2 User Scenario 2: Receive and Save Remediation Status and Results

Results of a remediation task are returned to the Remediation Manager from the Remediation Tool. If the remediation task has been performed, then the Remediation Manager updates the remediation task status to "accomplished." If a remediation task has not been performed and the task due time has passed, then the Remediation Manager updates the task status to "failed." If the remediation task is tagged as "failed," then the Remediation Manager creates a help desk ticket that describes the attempted remediation task and what failed.

A.2.3 User Scenario 3: Allow a User to Edit Policies, CRE/ERIs, Host Information, Remediation Tasks, Remediation Results, and Reports

The Remediation Manager allows a user to modify policies, CRE/ERIs, remediation tasks, remediation results, host information, and any reports that the Remediation Manager generates per User Scenario 5. The Remediation Manager saves all modifications along with the name of the user making the modification and the date of modification. Previous versions are also saved so that a history is maintained.

If a user modifies a policy that is higher than the local level of authority, then the Remediation Manager generates a POA&M that includes justification for the policy modification and any risks that are being accepted. POA&Ms are published in accordance with Netops data standards.

A.2.4 User Scenario 4: Allow a User to Assign Hosts and Policies to Policy Groups

The Remediation Manager allows a user to assign a host to a particular policy group. The user may modify this assignment at any time. The Remediation Manager allows a user to assign policies, which the user may modify at any time, to a particular policy group.

Newly discovered hosts will be placed into the unassigned group pending assignment by a user. The Remediation Manager sends an alert to notify the user when a new host is placed into a policy group.

A.2.5 User Scenario 5: Allow a User to Generate a Status Report

Periodically, the Remediation Manager automatically generates a remediation status report on all hosts in a given Remediation Tool's inventory; this report is generated according to the specified standard and provides information on all remediation tasks sent to that tool and their results and status along with the task due time. This report may be automatically sent to recipients who have requested that service. In addition, a manual report with the same information may be generated and displayed at any time by the user.

Furthermore, the Remediation Manager allows the user to manually generate reports on the content and hierarchy of policies as well as the health of all Remediation Manager components. The user may also generate an event log of all Remediation Manager actions over a given date range.

A.3 Remediation Manager High-Level Architecture

Figure 5 illustrates all of the subsystem components of the Remediation Manager. System-level requirements apply to the Remediation Manager. Subsystem-level requirements are decomposed from system requirements and apply to a subsystem component within the Remediation Manager. In Figure 5, all internal Remediation Manager subsystem components are illustrated in blue while external entities are in orange. The Remediation Manager system, subsystem components, and external entities are the following:

- CRE/ERI Store—persistent database for storing CRE/ERI data
- Findings Store—persistent database for storing issues (i.e., CCEs and CVEs) and their associated hosts, which are derived from the scan results
- Human-Machine Interface (HMI)—graphical interface between the other Remediation Manager components and a user
- Host Reader—Remediation Manager software component that receives host information and places it into the host store
- Host Store—persistent database for storing host information
- Listener—Remediation Manager software component that receives scan results and extracts information on issues (i.e., CCEs and CVEs) and the hosts on which they were found. This information is known as a finding.
- History Log—database or file for storing scan results, Remediation Manager component status, and event logs
- Policy Manager—Remediation Manager software component that manages hosts that have not been previously assigned a policy group. It contains the local policy editor, which allows a user to edit policies, policy groups, host information, and remediation tasks.
- Policy Store—persistent database for storing policy content, policy edits, and policy group information
- Report Generator—Remediation Manager software component that automatically generates remediation task results status and allows a user to generate reports on scan results, policies, hosts, CRE/ERI data, remediation tasks, remediation task results, Remediation Manager component status, and event logs
- Remediation Manager—the system being developed that is composed of all components shown in blue in Figure 5

- Remediation Tool—the tool that receives the remediation tasks from the Remediation Manager. It is not a component of the Remediation Manager but is an external entity shown in orange in Figure 5.
- Task Builder—Remediation Manager software component that gathers all of the data from the stores and uses that data to generate remediation tasks. It contains a number of subcomponents, two of which are the Administrator and Processor.
- Ticket Manager—an external entity that receives help desk tickets from the Remediation Manager. It is not a component of the Remediation Manager but is an external entity shown in orange in Figure 5.
- Task Store—persistent database for storing remediation tasks, remediation task results, and status
- Workflow Manager—Remediation Manager software component that gathers remediation tasks from the Task Store and packages them for output to the appropriate Remediation Tool

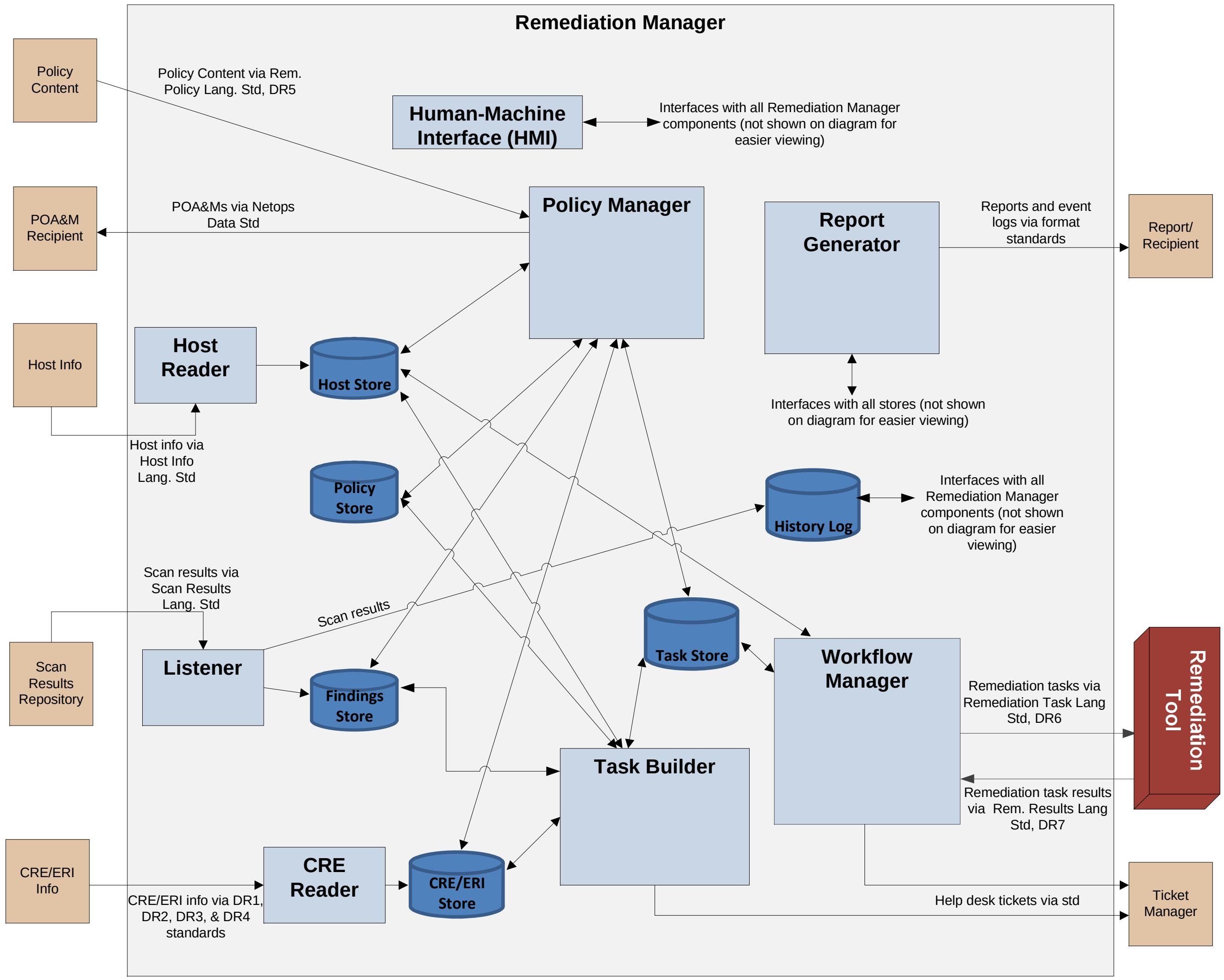


Figure 5: Remediation Manager High-Level Architecture

A.4 Standards

There are a number of existing and evolving standards with which the Remediation Manager must be in compliance. Table 7 lists the probable standards along with the expectations of the standard's content. Note that some of the emerging standards are referred to as "languages" and others as "formats." A language is generally a more complex expression than a format. It is possible that some of the standards identified in the table as languages will instead be specified as formats.

Table 7: Remediation Manager Standards [Waltermire 2011]

Standard	Std ID[9]	Standard Content
CRE	DR1	Standard way of uniquely identifying a remediation task. • Standard should express a definition for remediation tasks that includes parameter values in a predictable, parsable format. • Standard should include a mapping to CVEs and CCEs.
CRE-ML[10]	DR2	Standard definition of an exchange format for basic remediation information.
ERI	DR3	Standard definition of desired additional info about a remediation, including mappings to applicable platforms, related vulnerabilities, or configuration issues. • Standard should include a way of mapping CREs to ERIs.
ERI-ML	DR4	Standard definition of an expression language for the additional info about remediation identified in DR3.
Remediation Policy Language	DR5	Standard way of specifying which policies apply to which classes of assets (XML). • Standard should include a way of mapping particular policies to IT asset type. • Standard should include a way of uniquely defining asset types. • Standard should include a way of uniquely defining policy types (e.g., registry keys, file permissions). • Standard should define level(s) of readability for policy (e.g., by humans, by machine only). • Standard should include a way of defining dates in remediation policy and what dates (e.g., creation date, implementation date, and expiration date) are required or desired. • Standard should include criteria that can be used to select between multiple remediation options. • Standard should define how long assets may defer implementation of a remediation. • Standard should include info on who issued the policy, whom or what it applies to, if it is mandatory or optional, the policy issuer's authority or scope and—if multiple options exist—the order of preference. • Standard should include a way of stating who can send out policies, who can edit policies so that the Remediation Manager knows whom to accept policies from, and if they can be edited locally. • Standard should include a way of reporting policy groups to Remediation Tools and Remediation Managers and what policy groups are associated with which Remediation Tools. • Standard should include a definition for risk likelihood and impact and what level of risk may or may not be accepted by various IT assets.

9 DR stands for "Derived Requirement." DRs are identified in Table 1.

10 ML stands for metalanguage.

Table 7: Remediation Manager Standards [Waltermire 2011] (continued)

Standard	Std ID	Standard Content
Remediation Tasking Language	DR6	Standard way of applying specific remediation tasks to specific assets in an enterprise environment (XML). • Standard should include a way of mapping particular remediation tasks to IT asset type and/or Remediation Tool. • Standard should include a way of uniquely defining asset types and Remediation Tools. (It should match the Remediation Policy Language standard.) • Standard should express what remediation actions with what values will be performed on what assets via what tools. • Standard should define what remediation tasks are required, allowed, preferred, and/or prohibited. • Standard should include a way to express the order in which remediation tasks should be performed. • Standard should express what assets and Remediation Tools a Remediation Manager is allowed to task and what types of tasks those assets and Remediation Tools can support. • Standard should include a way for Remediation Tools and assets to know what Remediation Managers are allowed to task them and what tasks they may accept. • Standard should include a definition for risk likelihood and impact and what level of risk may or may not be accepted. (It should match the Remediation Policy Language.) • Standard should include a way of reporting policy groups to Remediation Tools and Remediation Managers and what policy groups are associated with which Remediation Tools. (It should match the Remediation Policy Language.)
Remediation Results Format	DR7	Standard way of reporting the results of an attempted remediation task. (Use a defined XML schema and follow the Remediation Manager to Remediation Tool Interface Control Document [ICD].) • Standard should define a way for Remediation Tools and assets to report back to the Remediation Manager what they did and did not do and why. • Standard should define a way of reporting exceptions to policy (POA&M) and to remediation tasks. • Standard should include a unique definition of error types (i.e., unsuccessful remediation tasks).
NA	DR8	Standard way of expressing how to perform a remediation task in a precise, machine-readable fashion. (Use a defined XML schema and follow the Remediation Manager to Remediation Tool ICD.) *[Note: DR 8 is not part of the work described herein and was rejected as a pursuit due to projected cost, complexity, likelihood of success, and lack of vendor support.]*
CCE	[MITRE 2011a]	Common Configuration Enumeration
CVE	[MITRE 2011b]	Common Vulnerabilities and Exposures
Scan Results Language		Standard way of reporting scan results. • Includes DoD ARF version 0.41 XML from ARCAT, XCCDF, ASR, OVAL • Standard should include a way of stating who can send out scan results so that the Remediation Manager knows whom to accept scan results from.
Active Directory API		Standard interface for tasking and reporting to and from Active Directory Services.
POA&M Format		Standard for POA&M format and content consistent with Netops Data Standards.
Patch Management Language		Standard definition for patch management.
Host Information Language		Standard way of reporting host information to the Remediation Manager and Remediation Tool.
ASR Report Format		Standard way for Remediation Managers to log what tasks they sent out, to what Remediation Tools/assets, on what authority, and based on what policy; current status of tasks; and what was reported back.
Remediation Results Report Format		Standard way for Remediation Tools to log what tasks they received, from whom they received the tasks, and what they did as a result.
Other SCAP content		Other content from the SCAP that has not already been identified.

A.5 System Requirements with Decomposition to Remediation Manager Subsystem Components

This section lists the System Requirements for the Remediation Manager along with their decomposition to the subsystem components within the Remediation Manager. The tables in this section provide the requirement level (system or subsystem component), a requirement identifier for ease of tracking, the text of the requirement, a reference/trace to a higher-level document for the requirement, and when the requirement is expected to be implemented. *Increment 1* was delivered in September 2010, *Increment 2* was delivered in December 2010, and *Future* is a future deliverable.

A.5.1 Standards and External ICDs

Requirement Level	Requirement ID	Requirement Text: Standards and External ICDs	Reference for Traceability[11]	Implementation Increment
Sys (Remediation Manager)	Remediation Manager 1.1	The Remediation Manager shall operate independently of any remediation actions and network Remediation Tools.	Integrated SOW	future
Sys (Remediation Manager)	Remediation Manager 1.2	The Remediation Manager shall be compliant with the following standards: • SCAP • CRE & CRE-ML (DR1 & DR2) • ERI & ERI-ML (DR3 & DR4) • Remediation Policy Language (DR5) • Remediation Tasking Language (DR6) • Remediation Results Format (DR7) • OVAL • CCE and CVE • ASR Report • Scan Results Language (DoD ARF version 0.41, XCCDF) • Remediation Tool Capability Language • Host Information Language • POA&M Format • Reports/History Log Format • Active Directory APIs	Remediation Manager 1.1	future
Sys (Remediation Manager)	Remediation Manager 1.3	The Remediation Manager shall be compliant with the following external ICDs: • Scan Results ICD • Remediation Policy Repository/Policy Input ICD • Remediation Tool to Remediation Manager ICD • Host/Asset Info ICD • CRE/ERI Input ICD • Help Desk Ticket ICD	Remediation Manager 1.1	future

[11] ISOW: U.S. National Security Agency. *Integrated Statement of Work for FY2010 Remediation Concept Development*. NSA, 2010.

A.5.2 Remediation Manager Inputs

Requirement Level	Requirement ID	Requirement Text: Remediation Manager Inputs	Reference for Traceability[12]	Implementation Increment
Sys (Remediation Manager)	Remediation Manager 2.1	The Remediation Manager shall accept scan results in DoD ARF version 0.41 consistent with the Scan Results Language Standard.	ISOW, user scenario 1	2
Sub (Listener)	Listener 2.1.1	The Listener shall accept scan results in DoD ARF version 0.41 consistent with the Scan Results Language Standard.	Remediation Manager 2.1	2
Sub (Listener)	Listener 2.1.2	The Listener shall save the scan results in the History Log.	Remediation Manager 2.1	future
Sub (Listener)	Listener 2.1.3	The Listener shall extract each host's noncompliant finding information (i.e., CCE or CVE) from the scan results and save that information in the Findings Store with a status of "new."	Remediation Manager 2.1	2
Sub (Findings Store)	Findings Store 2.1.1	The Findings Store shall store noncompliant finding information for each host in a persistent store.	Remediation Manager 2.1	2
Sub (History Log)	History Log 2.1.1	The History Log shall record scan results in the format they are received with a date/time stamp.	Remediation Manager 2.1	future
Sys (Remediation Manager)	Remediation Manager 2.2	The Remediation Manager shall accept policy instructions consistent with standards-Derived Requirement DR5, Remediation Policy Language, from a remediation policy repository.	ISOW, user scenario 1	future
Sub (Policy Manager)	Policy Manager 2.2.1	The Policy Manager shall accept policy content consistent with standards-Derived Requirement DR5, Remediation Policy Language.	Remediation Manager 2.2	future
Sub (Policy Manager)	Policy Manager 2.2.2	The Policy Manager shall save policy content in the Policy Store.	Remediation Manager 2.2	future
Sub (Policy Store)	Policy Store 2.2.1	The Policy Store shall store policy content in a persistent store.	Remediation Manager 2.2	2
Sys (Remediation Manager)	Remediation Manager 2.3	The Remediation Manager shall accept host information data consistent with the Host Information Language standard.	user scenario 1	future
Sub (Host Reader)	Host Reader 2.3.1	The Host Reader shall accept host information data consistent with the Host Information Language standard.	Remediation Manager 2.3	future
Sub (HT)	Host Reader 2.3.2	The Host Reader shall save host information data in the Host Store.	Remediation Manager 2.3	2
Sub (Host Store)	Host Store 2.3.1	The Host Store shall store host information data in a persistent store.	Remediation Manager 2.3	2

[12] ISOW: U.S. National Security Agency. *Integrated Statement of Work for FY2010 Remediation Concept Development*. NSA, 2010.

A.5.2 Remediation Manager Inputs (continued)

Requirement Level	Requirement ID	Requirement Text: Remediation Manager Inputs	Reference for Traceability	Implementation Increment
Sys (Remediation Manager)	Remediation Manager 2.4	The Remediation Manager shall accept CRE and ERI data consistent with the DR1, DR2, DR3, and DR4 standards.	user scenario 1	future
Sub (CRE Reader)	CRE Reader 2.4.1	The CRE Reader shall accept CRE data consistent with the DR1 and DR2 standards.	Remediation Manager 2.4	future
Sub (CRE Reader)	CRE Reader 2.4.2	The CRE Reader shall accept ERI data consistent with the DR3 and DR4 standards.	Remediation Manager 2.4	future
Sub (CRE Reader)	CRE Reader 2.4.3	The CRE Reader shall save CRE data in the CRE/ERI Store.	Remediation Manager 2.4	future
Sub (CRE Reader)	CRE Reader 2.4.4	The CRE Reader shall save ERI data in the CRE/ERI Store.	Remediation Manager 2.4	future
Sub (CRE/ERI Store)	CRE/ERI Store 2.4.1	The CRE/ERI Store shall store CRE data in a persistent store.	Remediation Manager 2.4	2
Sub (CRE/ERI Store)	CRE/ERI Store 2.4.2	The CRE/ERI Store shall store ERI data in a persistent store.	Remediation Manager 2.4	future
Sys (Remediation Manager)	Remediation Manager 2.5	The Remediation Manager shall accept results per the Remediation Results Format standard (standards-Derived Requirement DR7).	ISOW, user scenario 2	1
Sub (Task Store)	Task Store 2.5.1	The Task Store shall store remediation task result status in a persistent store.	Remediation Manager 2.5	2
Sub (Workflow Manager)	Workflow Manager 2.5.1	The Workflow Manager shall accept remediation task results per the Remediation Results Format standard (DR7).	Remediation Manager 2.5	1
Sub (Workflow Manager)	Workflow Manager 2.5.2	The Workflow Manager shall determine from the received results whether a remediation task has been accomplished, failed, or is in process.	Remediation Manager 2.5	1
Sub (Workflow Manager)	Workflow Manager 2.5.3	The Workflow Manager shall assign a remediation task a status of “accomplished” if that task has succeeded and a status of “failed” if that task has failed.	Remediation Manager 2.5	2
Sub (Workflow Manager)	Workflow Manager 2.5.4	The Workflow Manager shall update and save the status of a remediation task in the Task Store.	Remediation Manager 2.5	2
Sub (Workflow Manager)	Workflow Manager 2.5.5	If the status of a remediation task is “failed,” then the Workflow Manager shall generate a help desk ticket describing the attempted remediation task and why it failed.	Remediation Manager 2.5	2 (partial)

A.5.3 Remediation Manager Outputs

Requirement Level	Requirement ID	Requirement Text: Remediation Manager Outputs	Reference for Traceability[13]	Implementation Increment
Sys (Remediation Manager)	Remediation Manager 3.1	The Remediation Manager shall output a directive to apply a remediation task per standards-Derived Requirement DR6 - Remediation Tasking Language.	ISOW, user scenario 1	2
Sub (Workflow Manager)	Workflow Manager 3.1.1	The Workflow Manager shall send a directive to apply a remediation task per standards-Derived Requirement DR6, Remediation Tasking Language, to a particular Remediation Tool.	Remediation Manager 3.1	1
Sub (Workflow Manager)	Workflow Manager 3.1.2	The Workflow Manager shall retrieve the remediation task information including assigned Remediation Tool from the Task Store.	Remediation Manager 3.1	2
Sys (Remediation Manager)	Remediation Manager 3.2	The Remediation Manager shall publish remediation task results with notations on fixes made.	ISOW, user scenario 5	future
Sub (Report Generator)	Report Generator 3.2.1	The Report Generator shall generate and send a remediation status report on all hosts in a given Remediation Tool's inventory via an agreed to standard format.	Remediation Manager 3.2	future
Sub (Report Generator)	Report Generator 3.2.2	The Report Generator shall retrieve remediation task status from the Task Store.	Remediation Manager 3.2	future

13 ISOW: U.S. National Security Agency. *Integrated Statement of Work for FY2010 Remediation Concept Development*. NSA, 2010.

A.5.3 Remediation Manager Outputs (continued)

Requirement Level	Requirement ID	Requirement Text: Remediation Manager Outputs	Reference for Traceability	Implementation Increment
Sys (Remediation Manager)	Remediation Manager 3.3	The Remediation Manager shall output its event logs and reports to another system or service via an agreed-upon standard format.	user scenario 5	future
Sub (Report)	Report Generator 3.3.1	The Report Generator shall send its event logs and/or reports to another system or service via an agreed-upon standard format.	Remediation Manager 3.3	future
Sub (Report Generator)	Report Generator 3.3.2	The Report Generator shall retrieve reports from the History Log.	Remediation Manager 3.3	future
Sys (Remediation Manager)	Remediation Manager 3.4	The Remediation Manager shall output help desk tickets consistent with the appropriate standard to the appropriate system.	user scenarios 1 & 2	2
Sub (Task Builder)	Task Builder 3.4.1	The Task Builder shall send any help desk tickets it generates to the appropriate system consistent with the appropriate standard.	Remediation Manager 3.4	2
Sub (Workflow Manager)	Workflow Manager 3.4.1	The Workflow Manager shall send any help desk tickets it generates to the appropriate system consistent with the appropriate standard.	Remediation Manager 3.4	2
Sys (Remediation Manager)	Remediation Manager 3.5	The Remediation Manager shall publish POA&M messages consistent with Netops data standards.	ISOW, user scenario 3	future
Sub (Policy Manager)	Policy Manager 3.5.1	The Policy Manager shall publish POA&M messages consistent with Netops data standards.	Remediation Manager 3.5	future

A.5.4 User Interface and Functions

Requirement Level	Requirement ID	Requirement Text: User Interface and Functions	Reference for Traceability	Implementation Increment
Sys (Remediation Manager)	Remediation Manager 4.1	The Remediation Manager shall be managed via a graphical user interface.	user scenario 3, 4, and 5	future
Sub (HMI)	HMI 4.1.1	The HMI shall provide a graphical user interface to the user.	Remediation Manager 4.1	future
Sys (Remediation Manager)	Remediation Manager 4.2	The Remediation Manager shall allow users to choose which remediation to apply when multiple options are included in a policy.	ISOW, user scenario 1	future
Sub (HMI)	HMI 4.2.1	The HMI shall alert the user whenever a noncompliant finding has more than one potential remediation action that could be performed.	Remediation Manager 4.2	future
Sub (Task Builder)	Task Builder 4.2.1	The Task Builder shall create a help desk ticket whenever a noncompliant finding has more than one potential remediation action that could be performed.	Remediation Manager 4.2	2
Sub (Task Builder)	Task Builder 4.2.2	The Task Builder shall generate an alert to the HMI that notifies the user that there are more than one potential remediation actions against a single noncompliant finding.	Remediation Manager 4.2	future
Sys (Remediation Manager)	Remediation Manager 4.3	The Remediation Manager shall allow a user to tailor a policy for a given set of assets as well as accept some risks.	ISOW, user scenario 3	future
Sub (HMI)	HMI 4.3.1	The HMI shall be able to accept policy information from the Policy Manager and then graphically display that information.	Remediation Manager 4.3	future
Sub (HMI)	HMI 4.3.2	The HMI shall allow a user to edit policy information and then send any changes to the Policy Manager.	Remediation Manager 4.3	future
Sub (Policy Manager)	Policy Manager 4.3.1	The Policy Manager shall be able to accept policy information from the HMI.	Remediation Manager 4.3	future
Sub (Policy Manager)	Policy Manager 4.3.2	The Policy Manager shall allow users to edit policy information and provide a justification for accepting risks.	Remediation Manager 4.3	future

A.5.4 User Interface and Functions (continued)

Requirement Level	Requirement ID	Requirement Text: User Interface and Functions	Reference for Traceability	Implementation Increment
Sub (Policy Manager)	Policy Manager 4.3.3	The Policy Manager shall store policy information in the Policy Store.	Remediation Manager 4.3	future
Sub (Policy Manager)	Policy Manager 4.3.4	The Policy Manager shall retrieve policy information from the Policy Store.	Remediation Manager 4.3	future
Sub (Policy Store)	Policy Store 4.3.1	The Policy Store shall save policy information in a persistent store.	Remediation Manager 4.3	future
Sys (Remediation Manager)	Remediation Manager 4.4	The Remediation Manager shall allow users to create POA&Ms for policy deviations.	ISOW, user scenario 3	future
Sub (HMI)	HMI 4.4.1	The HMI shall accept POA&M information from the Policy Manager and then graphically display it.	Remediation Manager 4.4	future
Sub (HMI)	HMI 4.4.2	The HMI shall allow a user to input POA&M information on policy deviations and then send it to the Policy Manager.	Remediation Manager 4.4	future
Sub (Policy Manager)	Policy Manager 4.4.1	The Policy Manager shall accept user-created POA&Ms from the HMI.	Remediation Manager 4.4	future
Sub (Policy Manager)	Policy Manager 4.4.2	The Policy Manager shall store POA&Ms in the Policy Store.	Remediation Manager 4.4	future
Sub (Policy Manager)	Policy Manager 4.4.3	The Policy Manager shall retrieve POA&Ms from the Policy Store.	Remediation Manager 4.4	future
Sub (Policy Store)	Policy Store 4.4.1	The Policy Store shall save POA&Ms to a persistent store.	Remediation Manager 4.4	future
Sys (Remediation Manager)	Remediation Manager 4.5	The Remediation Manager shall be able to monitor and display the health of all the individual components of the Remediation Manager. This includes components' activities and error events. A User shall be able to view or print any log.	user scenario 5	future
Sub (CRE Reader)	CRE Reader 4.5.1	The CRE Reader shall send time-stamped data (e.g., activity, user, date, errors) on all its activities to the History Log.	Remediation Manager 4.5	future

A.5.4 User Interface and Functions (continued)

Requirement Level	Requirement ID	Requirement Text: User Interface and Functions	Reference for Traceability	Implementation Increment
Sub (HMI)	HMI 4.5.1	The HMI shall allow users to choose reports to be generated by the Report Generator and then graphically display or print report information from the Report Generator.	Remediation Manager 4.5	future
Sub (HMI)	HMI 4.5.2	The HMI shall send time-stamped data (e.g., activities, user, date, errors) on all its activities to the History Log.	Remediation Manager 4.5	future
Sub (Host Reader)	Host Reader 4.5.1	The Host Reader shall send time-stamped data (e.g., activity, user, date, errors) on all its activities to the History Log.	Remediation Manager 4.5	future
Sub (History Log)	History Log 4.5.1	The History Log shall save all log/event information to a persistent store.	Remediation Manager 4.5	1
Sub (Listener)	Listener 4.5.1	The Listener shall send time-stamped data (e.g., activity, user, data, and errors) on all its activities to the History Log.	Remediation Manager 4.5	future
Sub (Policy Manager)	Policy Manager 4.5.1	The Policy Manager shall send time-stamped data (e.g., activity, user, date, errors) on all its activities to the History Log.	Remediation Manager 4.5	future
Sub (Report Generator)	Report Generator 4.5.1	The Report Generator shall send time-stamped data (e.g., reports generated, user, date, errors) on all its activities to the History Log.	Remediation Manager 4.5	future
Sub (Report Generator)	Report Generator 4.5.2	The Report Generator shall retrieve information from the History Log and generate a report on the health and status of any Remediation Manager component and send that report to the HMI.	Remediation Manager 4.5	future
Sub (Task Builder)	Task Builder 4.5.1	The Task Builder shall send time-stamped data (e.g., tasks generated, user, date, errors) on all its activities to the History Log.	Remediation Manager 4.5	future
Sub (Workflow Manager)	Workflow Manager 4.5.1	The Workflow Manager shall send time-stamped data (e.g., remediation tasks sent, user, date, errors) on all its activities to the History Log.	Remediation Manager 4.5	1

A.5.4 User Interface and Functions (continued)

Requirement Level	Requirement ID	Requirement Text: User Interface and Functions	Reference for Traceability	Implementation Increment
Sys (Remediation Manager)	Remediation Manager 4.6	The Remediation Manager shall allow users to edit policies, CREs, ERIs, findings, host information, remediation tasks, remediation results, and reports.	user scenario 3	future
Sub (CRE Reader)	CRE Reader 4.6.1	The CRE Reader shall retrieve CRE/ERI information from the CRE/ERI Store and provide it to the HMI.	Remediation Manager 4.6	future
Sub (CRE Reader)	CRE Reader 4.6.2	The CRE Reader shall accept updated CRE/ERI information from the HMI and use it to update the appropriate fields in the CRE/ERI Store.	Remediation Manager 4.6	future
Sub (HMI)	HMI 4.6.1	The HMI shall graphically display information from any component in the Remediation Manager (e.g., CRE Reader, Host Reader, Listener, Policy Manager, Report Generator, Task Builder, and Workflow Manager).	Remediation Manager 4.6	future
Sub (HMI)	HMI 4.6.2	The HMI shall allow a user to edit information that is graphically displayed.	Remediation Manager 4.6	future
Sub (HMI)	HMI 4.6.3	The HMI shall send user-updated information back to the Remediation Manager component that provided it.	Remediation Manager 4.6	future
Sub (Host Reader)	Host Reader 4.6.1	The Host Reader shall retrieve host information from the Host Store and provide it to the HMI.	Remediation Manager 4.6	future
Sub (Host Reader)	Host Reader 4.6.2	The Host Reader shall accept updated host information from the HMI and use it to update the appropriate fields in the Host Store.	Remediation Manager 4.6	future
Sub (Listener)	Listener 4.6.1	The Listener shall retrieve noncompliant finding information from the Findings Store and provide it to the HMI.	Remediation Manager 4.6	future
Sub (Listener)	Listener 4.6.2	The Listener shall accept updated finding information from the HMI and use it to update the appropriate fields in the Findings Store.	Remediation Manager 4.6	future

A.5.4 User Interface and Functions (continued)

Requirement Level	Requirement ID	Requirement Text: User Interface and Functions	Reference for Traceability	Implementation Increment
Sub (Policy Manager)	Policy Manager 4.6.1	The Policy Manager shall retrieve policy information, including POA&Ms, from the Policy Store and provide it to the HMI.	Remediation Manager 4.6	future
Sub (Policy Manager)	Policy Manager 4.6.2	The Policy Manager shall accept updated policy information, including POA&Ms, from the HMI and use it to update the appropriate fields in the Policy Store.	Remediation Manager 4.6	future
Sub (Report Generator)	Report Generator 4.6.1	The Report Generator shall retrieve reports from the History Log and provide them to the HMI.	Remediation Manager 4.6	future
Sub (Report Generator)	Report Generator 4.6.2	The Report Generator shall accept updated report information from the HMI and use it to update the appropriate report in the History Log.	Remediation Manager 4.6	future
Sub (Task Builder)	Task Builder 4.6.1	The Task Builder shall retrieve task information, including status, from the Task Store and provide it to the HMI.	Remediation Manager 4.6	future
Sub (Task Builder)	Task Builder 4.6.2	The Task Builder shall accept updated task information, including status, from the HMI and use it to update the appropriate report in the Task Store.	Remediation Manager 4.6	future
Sys (Remediation Manager)	Remediation Manager 4.7	The Remediation Manager shall allow users to assign policies and hosts to policy groups.	user scenario 4	future
Sub (HMI)	HMI 4.7.1	The HMI shall graphically display policy group and hierarchy information from the Policy Manager.	Remediation Manager 4.7	future
Sub (HMI)	HMI 4.7.2	The HMI shall allow a user to assign a policy to a particular policy group.	Remediation Manager 4.7	future
Sub (HMI)	HMI 4.7.3	The HMI shall allow a user to assign a host to a particular policy group.	Remediation Manager 4.7	future
Sub (HMI)	HMI 4.7.4	The HMI shall send user-updated policy group assignments to the Policy Manager.	Remediation Manager 4.7	future
Sub (Policy Manager)	Policy Manager 4.7.1	The Policy Manager shall retrieve policy group information from the Policy Store and provide it to the HMI.	Remediation Manager 4.7	future

A.5.4 User Interface and Functions (continued)

Requirement Level	Requirement ID	Requirement Text: User Interface and Functions	Reference for Traceability	Implementation Increment
Sub (Policy Manager)	Policy Manager 4.7.2	The Policy Manager shall accept updated policy group information from the HMI and use it to update the appropriate fields in the Policy Store.	Remediation Manager 4.7	future
Sub (Policy Manager)	Policy Manager 4.7.3	The Policy Manager shall trace policies up and down the hierarchy and determine group inheritance and what policies take precedence and send this information to the HMI.	Remediation Manager 4.7	future
Sub (Policy Manager)	Policy Manager 4.7.4	The Policy Manager shall store policy hierarchy information in the Policy Store.	Remediation Manager 4.7	future
Sub (Policy Store)	Policy Store 4.7.1	The Policy Store shall save policy hierarchy information to a persistent store.	Remediation Manager 4.7	future
Sys (Remediation Manager)	Remediation Manager 4.8	The Remediation Manager shall allow a user to generate and view any report.	user scenario 5	future
Sub (HMI)	HMI 4.8.1	The HMI shall graphically display a report from the Report Generator.	Remediation Manager 4.8	future
Sub (HMI)	HMI 4.8.2	The HMI shall allow a user to select status on which remediation components to display.	Remediation Manager 4.8	future
Sub (HMI)	HMI 4.8.3	The HMI shall request a report from the Report Generator on the user-selected Remediation Manager component.	Remediation Manager 4.8	future
Sub (History Log)	History Log 4.8.1	The History Log shall save reports from the Report Generator to a persistent store.	Remediation Manager 4.8	future
Sub (Report Generator)	Report Generator 4.8.1	Upon request, the Report Generator shall send a particular report to the HMI.	Remediation Manager 4.8	future
Sub (Report Generator)	Report Generator 4.8.2	Upon request, the Report Generator shall retrieve noncompliant finding information from the Findings Store and generate a report.	Remediation Manager 4.8	future
Sub (Report Generator)	Report Generator 4.8.3	Upon request, the Report Generator shall retrieve policy information from the Policy Store and generate a report with the policy history and justification of any changes.	Remediation Manager 4.8	future

A.5.4 User Interface and Functions (continued)

Requirement Level	Requirement ID	Requirement Text: User Interface and Functions	Reference for Traceability	Implementation Increment
Sub (Report Generator)	Report Generator 4.8.4	Upon request, the Report Generator shall retrieve host information data from the Host Store and generate a report.	Remediation Manager 4.8	future
Sub (Report Generator)	Report Generator 4.8.5	Upon request, the Report Generator shall retrieve CRE/ERI data from the CRE/ERI Store and generate a report.	Remediation Manager 4.8	future
Sub (Report Generator)	Report Generator 4.8.6	Upon request, the Report Generator shall retrieve remediation task information, including status, from the Task Store and generate a report.	Remediation Manager 4.8	future
Sub (Report Generator)	Report Generator 4.8.7	Upon request, the Report Generator shall retrieve POA&M information from the Policy Store and generate a report of a POA&M that provides information on policies, deviation from policies, risks, and mitigations.	Remediation Manager 4.8	future
Sub (Report Generator)	Report Generator 4.8.8	Upon request, the Report Generator shall retrieve log information on a given Remediation Manager component from the History Log and generate a report on that component.	Remediation Manager 4.8	future
Sub (Report Generator)	Report Generator 4.8.9	The Report Generator shall store its reports in the History Log.	Remediation Manager 4.8	future

A.5.5 Remediation Manager Internal Functions

Requirement Level	Requirement ID	Requirement Text: Remediation Manager Internal Functions	Reference for Traceability	Implementation Increment
Sys (Remediation Manager)	Remediation Manager 5.1	The Remediation Manager shall periodically examine scan findings and policy inputs and then generate the appropriate remediation task for each host.	user scenario 1	2
Sub (Task Builder)	Task Builder 5.1.1	The Task Builder shall retrieve noncompliant finding information from the Findings Store.	Remediation Manager 5.1	2
Sub (Task Builder)	Task Builder 5.1.2	The Task Builder shall retrieve policy content from the Policy Store.	Remediation Manager 5.1	2
Sub (Task Builder)	Task Builder 5.1.3	The Task Builder shall retrieve host information data from the Host Store.	Remediation Manager 5.1	2
Sub (Task Builder)	Task Builder 5.1.4	The Task Builder shall retrieve CRE data from the CRE/ERI Store.	Remediation Manager 5.1	2
Sub (Task Builder)	Task Builder 5.1.5	The Task Builder shall retrieve ERI data from the CRE/ERI Store.	Remediation Manager 5.1	future
Sub (Task Builder)	Task Builder 5.1.6	The Task Builder shall retrieve POA&Ms from the Policy Store.	Remediation Manager 5.1	future
Sub (Task Builder)	Task Builder 5.1.7	The Task Builder shall determine the policy group of the host with findings against it, determine which policies apply to it, and generate a remediation task to address the finding.	Remediation Manager 5.1	2
Sub (Task Builder)	Task Builder 5.1.8	The Task Builder shall assign a status of “in process” to all newly generated remediation tasks.	Remediation Manager 5.1	2
Sub (Task Builder)	Task Builder 5.1.9	The Task Builder shall generate a help desk ticket if a particular finding has been found previously against the same host and the task due time has passed.	Remediation Manager 5.1	2
Sub (Task Builder)	Task Builder 5.1.10	The Task Builder shall update the status of all noncompliant findings in the Findings Store after a remediation task has been generated.	Remediation Manager 5.1	2

A.5.5 Remediation Manager Internal Functions (continued)

Requirement Level	Requirement ID	Requirement Text: Remediation Manager Internal Functions	Reference for Traceability	Implementation Increment
Sub (Task Builder)	Task Builder 5.1.11	Whenever the Task Builder finds that a host has not been assigned to a policy group, then the Task Builder shall generate an alert for the user to assign the host to a group.	Remediation Manager 5.1	future
Sub (Task Builder)	Task Builder 5.1.12	The Task Builder shall store remediation tasks in the Task Store.	Remediation Manager 5.1	2
Sub (Task Store)	Task Store 5.1.13	The Task Store shall save remediation task information in a persistent store.	Remediation Manager 5.1	2
Sys (Remediation Manager)	Remediation Manager 5.2	The Remediation Manager shall determine the appropriate Remediation Tool for each remediation task that has been generated.	ISOW, user scenario 1	2
Sub (Task Store)	Task Store 5.2.1	The Task Store shall save remediation task assignments to a persistent store.	Remediation Manager 5.2	2
Sub (Workflow Manager)	Workflow Manager 5.2.1	The Workflow Manager shall retrieve host information data from the Host Store.	Remediation Manager 5.2	2
Sub (Workflow Manager)	Workflow Manager 5.2.2	The Workflow Manager shall retrieve remediation task information from the Task Store.	Remediation Manager 5.2	2
Sub (Workflow Manager)	Workflow Manager 5.2.3	The Workflow Manager shall assign remediation tasks to the appropriate Remediation Tool based on the policy group to which a host is assigned and what tool is associated with it.	Remediation Manager 5.2	2
Sub (Workflow Manager)	Workflow Manager 5.2.4	The Workflow Manager shall store remediation task assignments in the Task Store.	Remediation Manager 5.2	2
Sys (Remediation Manager)	Remediation Manager 5.3	If the Remediation Manager does not receive a response from the Remediation Tool after a set period of time, then the Remediation Manager shall mark the task result as “failed” and shall generate a help desk ticket.	user scenario 2	future

A.5.5 Remediation Manager Internal Functions (continued)

Requirement Level	Requirement ID	Requirement Text: Remediation Manager Internal Functions	Reference for Traceability	Implementation Increment
Sub (Workflow Manager)	Workflow Manager 5.3.1	If the Workflow Manager does not receive a response from the Remediation Tool after a set period of time, then the Workflow Manager shall set the remediation task status to “failed” and generate a help desk ticket.	Remediation Manager 5.3	future

A.5.6 Remediation Manager Nonfunctional Requirements (Quality Attributes) and Miscellaneous

Requirement Level	Requirement ID	Requirement Text: Nonfunctional Requirements (Quality Attributes) and Miscellaneous	Reference for Traceability[14]	Implementation Increment
Sys (Remediation Manager)	Remediation Manager 6.1	The Remediation Manager shall be scalable and configurable for local as well as centralized management.	customer	future
Sys (Remediation Manager)	Remediation Manager 6.2	The Remediation Manager shall be capable of communicating via secure methods for downloading policy and content and providing report generation.	customer	future
Sys (Remediation Manager)	Remediation Manager 6.3	The Remediation Manager shall provide secure identification and authentication mechanisms between all components.	customer	future
Sys (Remediation Manager)	Remediation Manager 6.4	The Remediation Manager shall support DoD Public Key Infrastructure (PKI) certificates.	customer	future
Sys (Remediation Manager)	Remediation Manager 6.5	The Remediation Manager shall not interfere with the operation of DoD-mandated information assurance tools (e.g., antivirus).	customer	future
Sys (Remediation Manager)	Remediation Manager 6.6	The Remediation Manager shall be able to interface with third-party network operations tools (reporting, security information management systems, etc.).	customer	future
Sys (Remediation Manager)	Remediation Manager 6.7	The Remediation Manager shall ensure the integrity of stored data.	customer	future
Sys (Remediation Manager)	Remediation Manager 6.8	The Remediation Manager shall provide failover and/or redundancy capabilities.	customer	future
Sys (Remediation Manager)	Remediation Manager 6.9	The Remediation Manager shall be able to be run in a virtual environment.	customer	2
Sys (Remediation Manager)	Remediation Manager 6.10	The Remediation Manager shall provide integrity controls to protect against compromise of the remediation solution.	customer	future
Sys (Remediation Manager)	Remediation Manager 6.11	The Remediation Manager shall allow a user to configure role-based access controls.	customer	future

[14] These requirements are based on customer working draft materials that identify quality attribute requirements expected of an operational remediation management solution.

Appendix B Remediation Manager Reference Implementation Architecture and Design

B.1 Overview

This appendix documents the architectural views and the detailed design of the Remediation Manager. It focuses primarily on the second increment of the calendar year (CY) 2010 release of the Remediation Manager. References are made to subsequent releases where appropriate.

B.2 Architectural Views

This section lists the architectural views of the Remediation Manager. In particular, we focus on the following three views: use case (section B.2.1), object (section B.2.2), and component (section B.2.3).

B.2.1 Use Case View

This section defines all use cases that the Remediation Manager will target. Those with a release date of CY 2010 have been implemented in the CY 2010 release of the Remediation Manager. The remaining use cases are intended for subsequent Remediation Manager releases.

B.2.1.1 Assign Policies to Policy Groups

Reference #	1	**Description**	User assigns policies to policy groups using the policy editor tool.
Name	Assign Policies to Policy Groups		
Release	CY2011		
Actor(s)	end user		
Parent Use Case(s)	n/a		
Child Use Case(s)	n/a		

Precondition(s)

#	Description	And/Or
1	Policies exist in Policy Store.	and
2	Policy groups exist in Policy Store.	

Steps

#	Description	Actor(s)
1	Select policy.	user
2	Select policy group.	
3	Associate policy with group.	
4	Store association.	

Postcondition(s)

#	Description	And/Or
1	Policy is associated with policy group in policy database.	

B.2.1.2 Assign Host to Policy Groups

Reference #	2	**Description**	User assigns host to policy groups using the policy editor tool.
Name	Assign Host to Policy Groups		
Release	CY 2011		
Actor(s)	end user		
Parent Use Case(s)	n/a		
Child Use Case(s)	n/a		

Precondition(s)

#	Description	And/Or
1	Host exists in Host Store.	and
2	Policy groups exist in policy database.	

Steps

#	Description	Actor(s)
1	Select host.	User
2	Select policy group.	
3	Associate host with group.	
4	Store association.	

Postcondition(s)

#	Description	And/Or
1	Host is associated with policy group in policy database.	

B.2.1.3 Scan Hosts

Reference #	3	**Description**	Network scanners and sensors periodically scan hosts and send results (findings on hosts) in DoD ARF version 0.41 to the Remediation Manager. Upon receipt of a finding (scan result) in ARF, Remediation Manager stores finding in its Findings Store, marking it as “new.”
Name	Scan Hosts		
Release	CY 2010		
Actor(s)	network scanners and sensors, Remediation Manager		
Parent Use Case(s)	n/a		
Child Use Case(s)	n/a		

Precondition(s)

#	Description	And/Or
1	None.	

Steps

#	Description	Actor(s)
1	Scan host.	network scanners and sensors
2	Send findings (scan results) in DoD ARF version 0.41 to the Remediation Manager.	network scanners and sensors
3	Mark findings as "new."	Remediation Manager
4	Store findings in Findings Store.	Remediation Manager

Postcondition(s)

#	Description	And/Or
1	New findings (scan results) stored in Findings Store.	

B.2.1.4 Handle Findings

Reference #	4	**Description**	(This is an abstract use case. Its child uses cases are its concrete representations.) Remediation Manager periodically selects all new findings from its Findings Store. For each new finding, Remediation Manager (1) checks Findings Store to see if host associated with the finding has already been assigned to a policy group and (2) checks each finding to see if same issue has been encountered on same host before.
Name	Handle Findings		
Release	CY 2010		
Actor(s)	Remediation Manager		
Parent Use Case(s)	n/a		
Child Use Case(s)	5, 6, 7		

Precondition(s)

#	Description	And/Or
1	New findings in Findings Store.	

Steps

#	Description	Actor(s)
1	Select all new findings from Findings Store.	Remediation Manager
2	For each new finding, check Policy Store to see if host associated with finding has already been assigned to a policy group.	Remediation Manager
3	Check each new finding to see if same finding has been encountered on same host before.	Remediation Manager

Postcondition(s)

#	Description	And/Or
1	Differs for each child use case.	

B.2.1.5 Handle New Findings

Reference #	5	**Description**	If host has been assigned a policy group and issue has not been found on host before, send task to the Remediation Tool on host machine and set task status to “in process.”
Name	Handle New Findings		
Release	CY 2010		
Actor(s)	Remediation Manager		
Parent Use Case(s)	4		
Child Use Case(s)	n/a		

Precondition(s)

#	Description	And/Or
1	Host has been assigned to a policy group.	and
2	Issue has not been encountered on this host before.	

Steps

#	Description	Actor(s)
1	Create task for finding/host combination.	Remediation Manager
2	Set task status to “in process.”	Remediation Manager
3	Send task to Remediation Tool.	Remediation Manager

Postcondition(s)

#	Description	And/Or
1	Ticket sent to Remediation Tool.	and
2	Task status set to “in process.”	

B.2.1.6 Handle Unassigned Hosts

Reference #	6	**Description**	If host has not been assigned a policy group, create a task for Policy Manager indicating that the host must be assigned to a group.
Name	Handle Unassigned Hosts		
Release	CY 2010		
Actor(s)	Remediation Manager		
Parent Use Case(s)	4		
Child Use Case(s)	n/a		

Precondition(s)

#	Description	And/Or
1	Host has not been assigned to a policy group.	

Steps

#	Description	Actor(s)
1	If host record already exists in Host Store, go to step 3.	Remediation Manager
2	Create new host record in Host Store.	Remediation Manager
3	Create task indicating that host must be assigned to a group, and set task status to “in process” and target to Policy Manager.	Remediation Manager

Postcondition(s)

#	Description	And/Or
1	Host is present in Host Store.	and
2	Task created for group assignment of host.	and
3	Task status set to “in process” and target to Policy Manager.	

B.2.1.7 Handle Repeat Findings

Reference #	7	Description	If host has been assigned a policy group, an issue has been found on this host before, and due time for remediation has passed, create task with status “failed” and target Ticket Manager.
Name	Handle Repeat Findings		
Release	CY 2010		
Actor(s)	Remediation Manager		
Parent Use Case(s)	4		
Child Use Case(s)	n/a		

Precondition(s)

#	Description	And/Or
1	Host has been assigned to a policy group.	and
2	Issue has been encountered on this host before.	and
3	Due time for remediation has passed.	

Steps

#	Description	Actor(s)
1	Create task for finding.	Remediation Manager
2	Set task status to “failed” and target to Ticket Manager.	Remediation Manager

Postcondition(s)

#	Description	And/Or
1	Task created to remedy.	and
2	Task status set to “failed” and target to Ticket Manager.	

B.2.1.8 Process Failed Tasks

Reference #	8	**Description**	When Remediation Tool informs Remediation Manager that a task has failed, set status to “failed” and send ticket to Ticket Manager.
Name	Process Failed Tasks		
Release	CY 2010		
Actor(s)	Remediation Manager Remediation Tool		
Parent Use Case(s)	n/a		
Child Use Case(s)	n/a		

Precondition(s)

#	Description	And/Or
1	Remediation task has been sent to Remediation Tool.	and
2	Remediation task has failed.	

Steps

#	Description	Actor(s)
1	Send Remediation Manager notification of failed task.	Remediation Tool
2	Send ticket to Ticket Manager.	Remediation Manager
3	Set task status to “failed” and target to Ticket Manager.	Remediation Manager

Postcondition(s)

#	Description	And/Or
1	Task sent to Ticket Manager.	and
2	Task status set to “failed” and target to Ticket Manager.	

B.2.1.9 Process Successful Tasks

Reference #	9	**Description**	When Remediation Tool informs Remediation Manager that a task has succeeded, set status to “accomplished.”
Name	Process Successful Tasks		
Release	CY 2010		
Actor(s)	Remediation Manager Remediation Tool		
Parent Use Case(s)	n/a		
Child Use Case(s)	n/a		

Precondition(s)

#	Description	And/Or
1	Remediation task has been sent to Remediation Tool.	and
2	Remediation task has succeeded.	

Steps

#	Description	Actor(s)
1	Send Remediation Manager notification of successful task.	Remediation Tool
2	Set task status to “accomplished.”	Remediation Manager

Postcondition(s)

#	Description	And/Or
1	Task status set to “accomplished.”	

B.2.1.10 Print Remediation Report

Reference #	10	**Description**	User may generate report on remediation status of all machines in Remediation Manager’s inventory.
Name	Print Remediation Report		
Release	CY 2011		
Actor(s)	user		
Parent Use Case(s)	n/a		
Child Use Case(s)	n/a		

Precondition(s)

#	Description	And/Or
1	None.	

Steps

#	Description	Actor(s)
1	Generate report.	user
2	Print report.	user

Postcondition(s)

#	Description	And/Or
1	Report is printed.	

B.2.1.11 Auto-Generate Remediation Report

Reference #	11	**Description**	Remediation Manager will periodically generate report on remediation status of all machines in Remediation Manager’s inventory and make report available to users.
Name	Auto-Generate Remediation Report		
Release	CY 2011		
Actor(s)	Remediation Manager		
Parent Use Case(s)	n/a		
Child Use Case(s)	n/a		

Precondition(s)

#	Description	And/Or
1	None.	

Steps

#	Description	Actor(s)
1	Generate report.	Remediation Manager
2	Deposit generated report for user pickup.	Remediation Manager

Postcondition(s)

#	Description	And/Or
1	Report deposited for user pickup.	

B.2.2 Object View

In this section, we focus on the object view of the Remediation Manager. This view is specified in terms of the class model (Section B.2.2.1) showing various objects and their interrelationships, state diagrams (Section B.2.2.2) describing the life cycle of various objects, and a flowchart (Section B.2.2.3) showing how the life cycles of various objects fit together within the overall logic of the Remediation Manager.

B.2.2.1 Class Model

Figure 6 shows the major classes in the design of the Remediation Manager and the relationships between them.

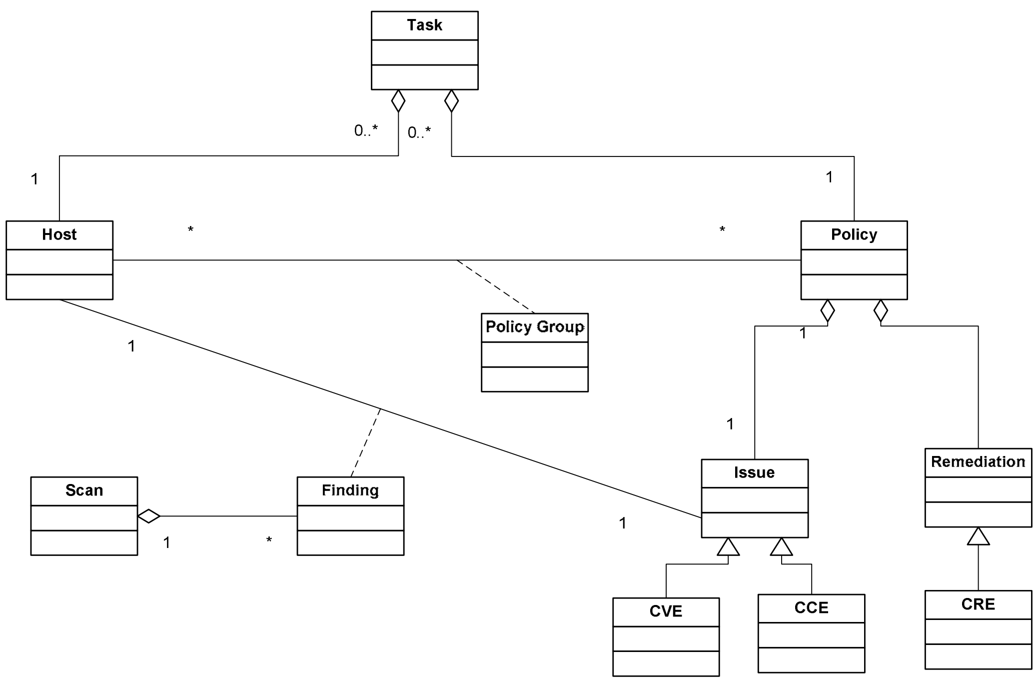


Figure 6: Remediation Manager Class Model

B.2.2.2 State Diagrams

This section shows state diagrams for various types of objects.

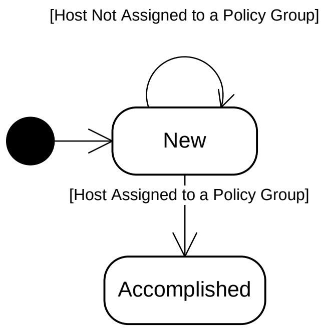


Figure 7: States for a "Finding" Object

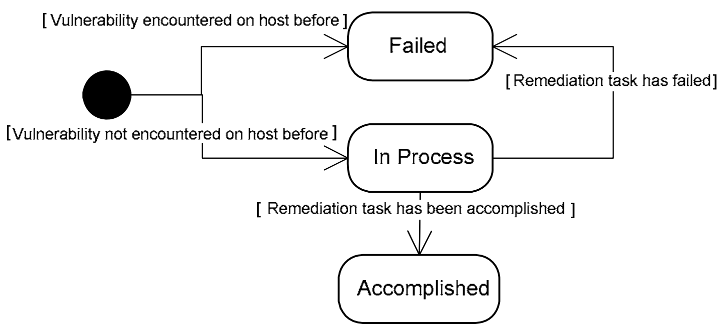


Figure 8: States for a "Task" Object

B.2.2.3 Flow Chart

Figure 9 is the flow chart for the Remediation Manager logic. The CY 2010 implementation does not include all the steps in this diagram: The policy manager subsystem component has not been developed, so the processing step “Policy Manager assigns host to policy group” is not implemented.

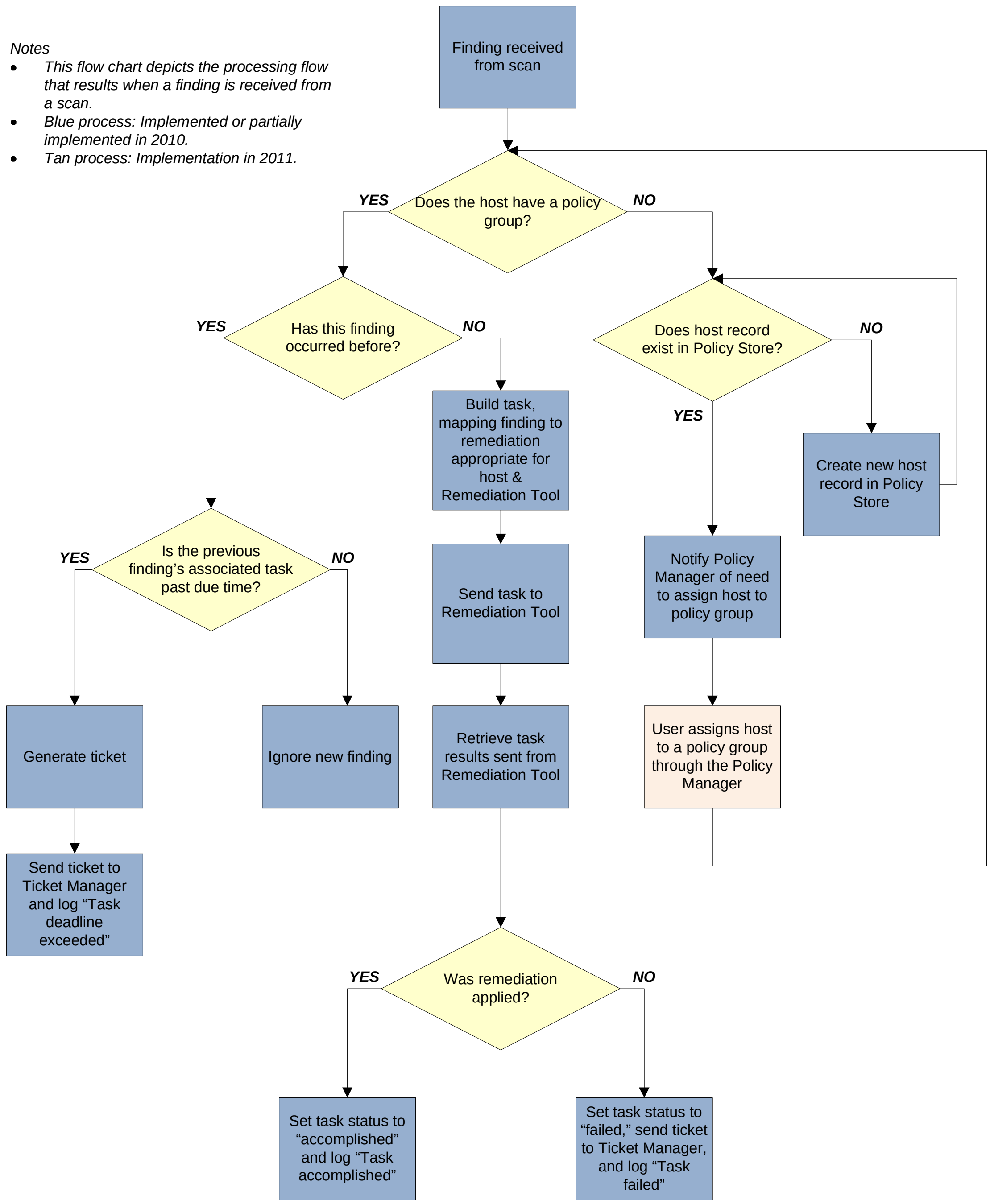


Figure 9: Remediation Manager Flow Chart

B.2.3 Component View

Figure 4 in Chapter 3 shows the component view of the Remediation Manager. It also appears in Appendix A (see Figure 5). For information on the standards referenced in this figure, see Appendix A, Section 4.

B.3 Detailed Design

In this section, we present the detailed design of the Remediation Manager. We focus on the software components created in CY 2011, the interfaces, and the data store.

B.3.1 Software Components

The CY 2010 version of the Remediation Manager is composed of three subsystem components, the Task Builder, the Workflow Manager, and the Listener. The Task Builder is further internally partitioned into two subcomponents, the Administrator and the Processor. Additional Remediation Manager subsystem components are planned but have not yet been implemented.

B.3.1.1 Task Builder

B.3.1.1.1 Administrator

This subcomponent of the Task Builder allows users to manage all the Remediation Manager components and subcomponents, specifically in the CY 2010 version, to start, pause, and resume the Listener, the Processor, and the Workflow Manager.

B.3.1.1.2 Processor

This subcomponent of the Task Builder periodically examines the findings in the Findings Store and assigns tasks to new findings. The logic employed in assigning tasks is as depicted in Figure 9.

B.3.1.2 Workflow Manager

This subsystem component of the Remediation Manager periodically examines tasks in the database and sends them to the Remediation Tool. This component also creates tasks for the Ticket Manager (e.g., Remedy) and the Policy Manager, but the Ticket Manager and Policy Manager do not yet exist in the implemented system.

B.3.1.3 Listener

This subsystem component of the Remediation Manager is a server that listens to a socket for ARF data (DoD ARF version 0.41). Upon receipt of an ARF document, the Listener extracts the issues found (CVEs and CCEs) and the hosts on which they were found. These are recorded as new findings in the Findings Store.

B.3.2 Interfaces

In the CY 2010 version, the Remediation Manager interfaces with two external entities:

- Remediation Tool—This entity executes the remediations tasked by the Remediation Manager. The Remediation Manager sends tasks to the Remediation Tool in Remediation

Tasking Language (RTL) format. The Remediation Tool also returns task results back to the Remediation Manager in Remediation Results Format (RRF). Appendix A, Section 4, describes RTL and RRF.

- Host Scanner—This entity scans hosts and sends records of vulnerabilities and improper configurations to the Remediation Manager (specifically, the Listener) in ARF (DoD ARF version 0.41).

B.3.2.1 Interface between Remediation Manager and Remediation Tool

The Remediation Manager interacts with the Remediation Tool using an asynchronous client-server model. Specifically, the Remediation Manager acts as the server and listens for incoming connections on a designated TCP/IP port. The IP address and port number of the Remediation Manager is known to the Remediation Tool. At its discretion, the Remediation Tool connects with the Remediation Manager on the designated port and participates in one of the following two possible interactions:

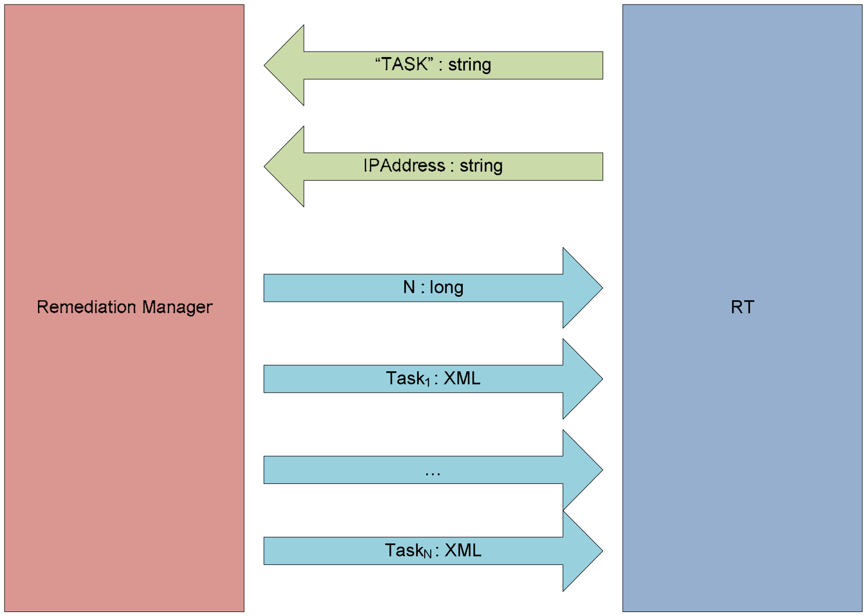
1. task interaction—The Remediation Tool obtains a set of remediation tasks from the Remediation Manager.
2. result interaction—The Remediation Tool returns a set of remediation results back to the Remediation Manager.

We now describe the message sequences involved in each of these two interactions. Each message is designated as “Name:Type.” The type is either “long,” “string,” or “XML.” Messages of different types are transmitted (and received) as follows:

- A long message is transmitted as a sequence of 8 bytes representing the signed 2’s complement value of the message, in order from the least to the most significant byte.
- A string message is transmitted by first sending the number of bytes in the string as a long message (as described in the previous item) and then the actual bytes in the string from the first to the last.
- An XML message is transmitted by first sending the number of bytes in the corresponding XML object’s string representation as a long message and then the actual bytes in the string representation from the first to the last.

B.3.2.1.1 Task Interaction

A task interaction involves the following sequence of message exchanges:



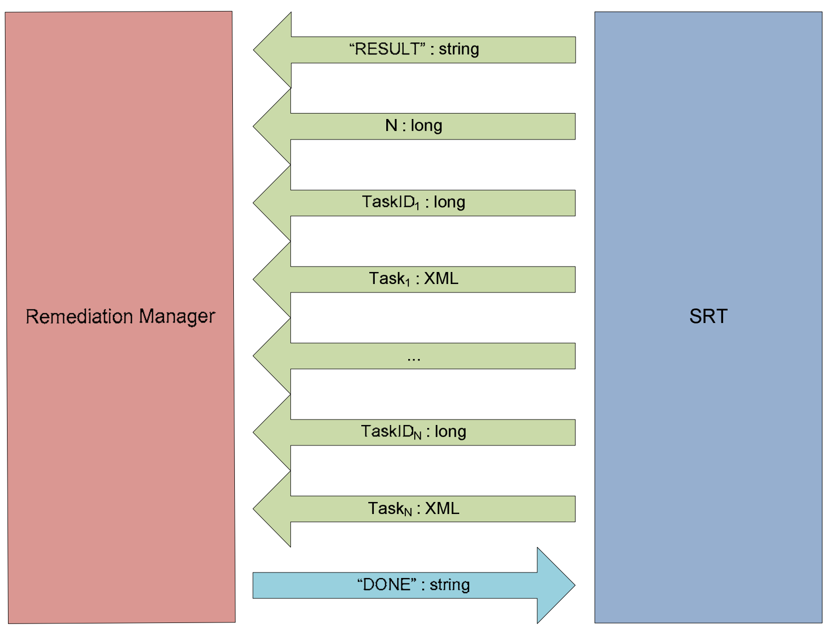
Notes:

1. IPAddress is the IP address of the remediation task target of interest to the Remediation Tool.
2. N is the number of remediation tasks sent by the Remediation Manager to the Remediation Tool.
3. $Task_i$ is a remediation tasking object from the schema presented in Section B.4.

Figure 10: Task Interaction

B.3.2.1.2 Result Interaction

A result interaction involves the following sequence of message exchanges:



Notes:

1. N is the number of remediation results sent by the Remediation Manager to the Remediation Tool.
2. $TaskId_i$ is the ID of the task that $Result_i$ corresponds to.
3. $Result_i$ is a Remediation Result object from the schema presented in Section B.4.

Figure 11: Result Interaction

B.3.2.2 Assessment Results Format (ARF) Interface

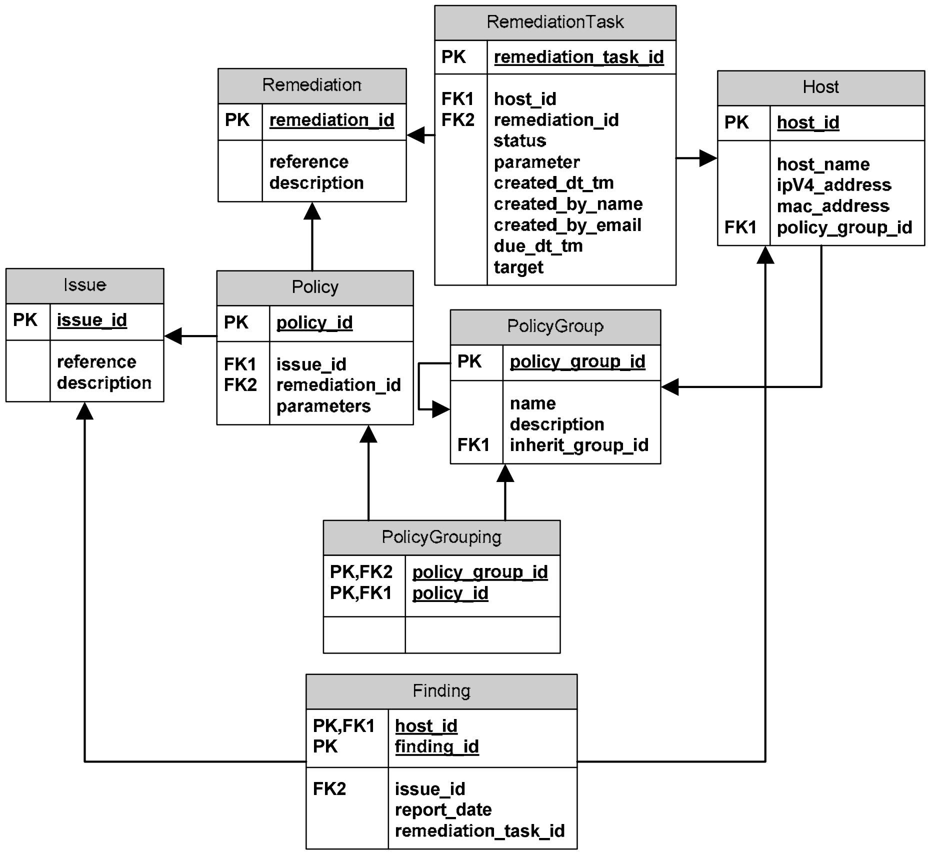
This standard XML interface is used by a tool that scans hosts for configuration errors or vulnerabilities to share scan findings with the Remediation Manager. Although the DoD ARF version 0.41 specification is large, the Remediation Manager is concerned only with a small subset of the data.

B.3.3 Data Store

In the current architecture, the Workflow Manager, Task Builder, and Policy Manager share a *single physical* data store hosted on an instance of MySQL. (The Policy Manager has not yet been developed, but the intent is for it to use this store as well). The physical data store encompasses the following logical stores indicated in the component diagram in Figure 4:

- Findings Store—This store contains the results of scans (pairings of hosts and CVEs or CCEs) submitted to the Task Builder by network scanners.
- Host Store—This store contains a list of the hosts whose remediation is monitored by the Remediation Manager.
- CRE Store—This store contains a list of remediations (CREs) applicable to the hosts whose remediations are managed by the Remediation Manager.
- Task Store—This store contains a list of tasks created by the Task Builder and sent to the Workflow Manager.
- Policy Store—This store contains policy groups, which associate hosts and policies. Policies, in turn, are associations between issues (CVEs and CCEs) and remediations (CREs). Policies are also stored in the Policy Store.

Figure 12 shows a semiphysical data model for this data store.



Note: PK = primary key, FK = foreign key

Figure 12: Remediation Manager Data Store for Remediation Tasks and Results

B.4 XML Schema for Remediation Tasks and Results

This section contains the XML schema respected by remediation tasks and remediation results.

```
<xs:schema targetNamespace=“http://www.sei.cmu.edu/remediation/wfm”
 elementFormDefault=“qualified” attributeFormDefault=“unqualified” xml:lang=“en”
 xmlns:wfm=“http://www.sei.cmu.edu/remediation/wfm”
 xmlns:xs=“http://www.w3.org/2001/XMLSchema”>
 <xs:element name=“remediationTasking”>
  <xs:complexType>
   <xs:sequence>
    <!-- This identifies the Remediation Tool that is expected to receive and process this document. -->
    <!-- What’s actually needed to identify a host will need to be decided. -->
    <xs:element name=“remediationTaskingTarget”
          type=“wfm:remediationTaskingTargetType” minOccurs=“1”/>

    <!-- This identifies the Remediation Manager that created and issued this document. -->
    <xs:element name=“remediationTaskingSource”
          type=“wfm:remediationTaskingSourceType”/>

    <!-- Information about the tasking document as a whole: who created it and when,
        when should it be completed or acknowledged, etc. -->
    <xs:element name=“remediationTaskingMetadata”
          type=“wfm:remediationTaskingMetadataType”/>

    <!-- The set of tasks which the Remediation Tool is expected to carry out. -->
    <xs:element name=“remediationTasks”
```

```
            type="wfm:remediationTasksType"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

<xs:element name="remediationResult">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="singleRemediationResult"
            type="wfm:singleRemediationResultType"
                        minOccurs="1" maxOccurs="1"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

<xs:complexType name="singleRemediationResultType">
  <xs:sequence>
    <!-- id of corresponding remediation task -->
    <xs:element name="remediationTaskId" type="wfm:taskIdType"/>

    <!-- id of corresponding remediation action -->
    <xs:element name="remediationActionId" type="wfm:actionIdType"/>

    <!-- Remediation result value -->
    <xs:element name="remediationResultValue"
           type="wfm:remediationResultValueType"/>

    <!-- Remediation result reason -->
    <xs:element name="remediationResultReason"
                        type="wfm:remediationResultReasonType"/>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="remediationTaskingTargetType">
  <xs:sequence>
    <xs:element name="hostName" type="wfm:targetHostNameType"/>
    <xs:element name="ipV4Address" type="wfm:ipV4AddressType"/>
    <xs:element name="macAddress" type="wfm:macAddressType"/>
    <xs:element name="applicationName" type="wfm:targetApplicationNameType"/>
  </xs:sequence>
</xs:complexType>

<xs:simpleType name="targetHostNameType">
  <xs:restriction base="xs:string">
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="targetApplicationNameType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="SRT"/>
  </xs:restriction>
</xs:simpleType>

<xs:complexType name="remediationTaskingSourceType">
  <xs:sequence>
    <xs:element name="hostName" type="wfm:sourceHostNameType"/>
    <xs:element name="ipV4Address" type="wfm:ipV4AddressType"/>
    <xs:element name="macAddress" type="wfm:macAddressType"/>
    <xs:element name="applicationName" type="wfm:sourceApplicationNameType"/>
  </xs:sequence>
</xs:complexType>

<xs:simpleType name="sourceHostNameType">
  <xs:restriction base="xs:string">
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="sourceApplicationNameType">
```

```
    <xs:restriction base=“xs:string”>
    </xs:restriction>
  </xs:simpleType>

  <xs:complexType name=“remediationTaskingMetadataType”>
    <xs:sequence>
      <xs:element name=“createddateTime” type=“wfm:dateTimeType”/>
      <xs:element name=“createdByName” type=“wfm:createdByNameType”/>
      <xs:element name=“createdByEmail” type=“wfm:emailAddressType”/>
      <xs:element name=“duedateTime” type=“wfm:dateTimeType”/>
    </xs:sequence>
  </xs:complexType>

  <xs:simpleType name=“createdByNameType”>
    <xs:restriction base=“xs:string”>
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name=“remediationResultValueType”>
    <xs:restriction base=“xs:string”>
      <xs:enumeration value=“SUCCESS”/>
      <xs:enumeration value=“FAILURE”/>
      <xs:enumeration value=“NO_HOST”/>
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name=“remediationResultReasonType”>
    <xs:restriction base=“xs:string”>
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name=“dateTimeType”>
    <xs:restriction base=“xs:string”>
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name=“ipV4AddressType”>
    <xs:restriction base=“xs:string”>
      <xs:pattern value=“((25[0-5]|2[0-4][0-9]|1[0-9][0-9]|[1-9][0-9]|[0-9])\.){3}(25[0-5]|2[0-4][0-9]|1[0-9][0-9]|[1-9][0-9]|[0-
9])”/>
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name=“macAddressType”>
    <xs:restriction base=“xs:string”>
      <xs:pattern value=“(([0-9]|[a-f]|[A-F])([0-9]|[a-f]|[A-F]):){5}(([0-9]|[a-f]|[A-F])([0-9]|[a-f]|[A-F]))”/>
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name=“emailAddressType”>
    <xs:restriction base=“xs:string”>
    </xs:restriction>
  </xs:simpleType>

  <xs:complexType name=“remediationTasksType”>
    <xs:sequence>
      <xs:element name=“remediationTask” type=“wfm:remediationTaskType” minOccurs=“1”/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name=“remediationTaskType”>
    <xs:sequence>
      <!-- id for tracking -->
      <xs:element name=“id” type=“wfm:taskIdType”/>
      <xs:element name=“remediationActions” type=“wfm:remediationActionsType”/>
      <xs:element name=“remediationTargets” type=“wfm:remediationTargetsType”/>
    </xs:sequence>
  </xs:complexType>
```

```
<xs:simpleType name="taskIdType">
 <xs:restriction base="xs:string">
 </xs:restriction>
</xs:simpleType>

<xs:complexType name="remediationActionsType">
 <xs:sequence>
  <xs:element name="remediationAction" type="wfm:remediationActionType" maxOccurs="unbounded"/>
 </xs:sequence>
</xs:complexType>

<xs:complexType name="remediationTargetsType">
 <xs:sequence>
  <xs:element name="remediationTarget" type="wfm:remediationTargetType" maxOccurs="unbounded" />
 </xs:sequence>
</xs:complexType>

<xs:complexType name="remediationActionType">
 <xs:sequence>
  <!-- id for tracking -->
  <xs:element name="id" type="wfm:actionIdType"/>
  <xs:element name="reference" type="wfm:actionReferenceType"/>
  <xs:element name="description" type="wfm:actionDescType"/>
  <xs:element name="parameter" type="wfm:paramType"/>
 </xs:sequence>
</xs:complexType>

<xs:simpleType name="actionIdType">
 <xs:restriction base="xs:string">
 </xs:restriction>
</xs:simpleType>

<xs:simpleType name="actionReferenceType">
 <xs:restriction base="xs:string">
 </xs:restriction>
</xs:simpleType>

<xs:simpleType name="actionDescType">
 <xs:restriction base="xs:string">
 </xs:restriction>
</xs:simpleType>

<xs:complexType name="paramType">
 <xs:sequence>
  <xs:element name="parameterType" type="wfm:parameterTypeType"/>
  <xs:element name="parameterValue" type="wfm:parameterValueType"/>
 </xs:sequence>
 <xs:attribute name="name" type="wfm:paramNameType" use="required"/>
</xs:complexType>

<xs:simpleType name="paramNameType">
 <xs:restriction base="xs:string">
 </xs:restriction>
</xs:simpleType>

<xs:simpleType name="parameterTypeType">
 <xs:restriction base="xs:string">
 </xs:restriction>
</xs:simpleType>

<xs:complexType name="parameterValueType">
 <xs:choice>
  <xs:element name="registry_state" type="wfm:registry_stateType"/>
 </xs:choice>
</xs:complexType>

<xs:complexType name="registry_stateType">
```

```
  <xs:sequence>
    <xs:element name=“hive” type=“wfm:regeditHiveType”/>
    <xs:element name=“key” type=“wfm:regeditKeyType”/>
    <xs:element name=“name” type=“wfm:regeditNameType”/>
    <xs:element name=“type” type=“wfm:regeditTypeType”/>
    <xs:element name=“value” type=“wfm:regeditValueType”/>
  </xs:sequence>
</xs:complexType>

<xs:simpleType name=“regeditHiveType”>
  <xs:restriction base=“xs:string”>
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name=“regeditKeyType”>
  <xs:restriction base=“xs:string”>
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name=“regeditNameType”>
  <xs:restriction base=“xs:string”>
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name=“regeditTypeType”>
  <xs:restriction base=“xs:string”>
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name=“regeditValueType”>
  <xs:restriction base=“xs:string”>
  </xs:restriction>
</xs:simpleType>

<xs:complexType name=“filePermissionsType”>
  <xs:sequence>
    <xs:element name=“filepath” type=“xs:string”/>
    <xs:element name=“path” type=“xs:string”/>
    <xs:element name=“filename” type=“xs:string”/>
    <xs:element name=“username” type=“xs:string”/>
    <xs:element name=“trustee_sid” type=“xs:string”/>
    <xs:element name=“standard_delete” type=“xs:boolean”/>
    <xs:element name=“standard_read_control” type=“xs:boolean”/>
    <xs:element name=“standard_write_dac” type=“xs:boolean”/>
    <xs:element name=“standard_writer_owner” type=“xs:boolean”/>
    <xs:element name=“standard_sync” type=“xs:boolean”/>
    <xs:element name=“access_sys_sec” type=“xs:boolean”/>
    <xs:element name=“generic_read” type=“xs:boolean”/>
    <xs:element name=“generic_write” type=“xs:boolean”/>
    <xs:element name=“generic_execute” type=“xs:boolean”/>
    <xs:element name=“generic_all” type=“xs:boolean”/>
    <xs:element name=“file_read_data” type=“xs:boolean”/>
    <xs:element name=“file_write_data” type=“xs:boolean”/>
    <xs:element name=“file_append_data” type=“xs:boolean”/>
    <xs:element name=“file_read_ea” type=“xs:boolean”/>
    <xs:element name=“file_write_ea” type=“xs:boolean”/>
    <xs:element name=“file_execute” type=“xs:boolean”/>
    <xs:element name=“file_delete_child” type=“xs:boolean”/>
    <xs:element name=“file_read_attributes” type=“xs:boolean”/>
    <xs:element name=“file_write_attributes” type=“xs:boolean”/>
  </xs:sequence>
</xs:complexType>

<xs:complexType name=“auditPolicyType”>
  <xs:sequence>
    <xs:element name=“System” type=“xs:string”/>
    <xs:element name=“Logon” type=“xs:string”/>
    <xs:element name=“ObjectAccess” type=“xs:string”/>
```

```
      <xs:element name="PrivilegeUse" type="xs:string"/>
      <xs:element name="DetailedTracking" type="xs:string"/>
      <xs:element name="PolicyChange" type="xs:string"/>
      <xs:element name="AccountManagement" type="xs:string"/>
      <xs:element name="DirectoryServiceAccess" type="xs:string"/>
      <xs:element name="AccountLogon" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="accountLockoutPolicyType">
    <xs:sequence>
      <xs:element name="force_logoff" type="xs:integer"/>
      <xs:element name="lockout_duration" type="xs:integer"/>
      <xs:element name="lockout_observation_window" type="xs:integer"/>
      <xs:element name="lockout_threshold" type="xs:integer"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="remediationTargetType">
    <xs:sequence>
      <xs:element name="id" type="wfm:remediationTargetIdType"/>
      <xs:element name="hostName" type="wfm:hostNameType"/>
      <xs:element name="ipV4Address" type="wfm:ipV4AddressType"/>
      <xs:element name="macAddress" type="wfm:macAddressType"/>
    </xs:sequence>
  </xs:complexType>

  <xs:simpleType name="remediationTargetIdType">
    <xs:restriction base="xs:string">
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="hostNameType">
    <xs:restriction base="xs:string">
    </xs:restriction>
  </xs:simpleType>

</xs:schema>
```

Appendix C Acronym List

ARCAT	Assessment Results Consumer & Analysis Tool, software system reference implementation to realize, demonstrate, and promote the use of the Assessment Results Format (ARF) and other Data Exchange Standards (DES)[15]
ARF	Assessment Results Format, an XML-based data exchange standard developed from Net D schemas for describing assessment results grouped by device[16]
ASR	Assessment Summary Results, a data exchange standard for describing assessment results grouped by individual findings[17]
CCE	Common Configuration Enumeration (CCE™) [MITRE 2011b]
CRE	Common Remediation Enumeration. A CRE entry is a set of actions taken to remediate a vulnerability or misconfiguration on a host. The enumerated list of all standardized CREs is itself referred to as the CRE [Waltermire 2011, p. 5].
CVE	Common Vulnerabilities and Exposures (CVE®) [MITRE 2011a]
DR	Derived Requirement
ERI	Extended Remediation Information
ICD	Interface Control Document
JDBC	Java Database Connectivity
NIST CSD	National Institute of Standards and Technology Computer Security Division
NVD	National Vulnerability Database
OVAL	Open Vulnerability and Assessment Language
POA&M	Plan of Action and Milestones
RFI	Request for Information
RRF	Remediation Results Format (also known as Remediation Results, Remediation Results Language, and Remediation Tasking Results) [Waltermire 2011, p. 3]
RTL	Remediation Tasking Language (formerly Remediation Control Language) [Waltermire 2011, p. 8]
SCAP	Security Content Automation Protocol
SOW	Statement of Work
SRT	SPAWAR Remediation Tool
XCCDF	eXtensible Configuration Checklist Description Format, a specification language for writing security checklists, benchmarks, and related kinds of documents

15 U.S. Department of Defense. *Software Requirements Specification, Assessment Results Consumer & Analysis Tool (ARCAT) Spiral Two.* November 6, 2009.

16 U.S. Department of Defense. *Assessment Results Format XML Specification, version 0.41*. http://metadata.dod.mil/mdr/ns/netops/shared_data/arf_index_page/0.41 (sponsored access required) 2010.

17 U.S. Department of Defense. *Assessment Summary Results Format v 0.41 draft.* September 12, 2009.

Bibliography

URLs are valid as of the publication date of this document.

[MITRE 2011a]
MITRE. *Common Vulnerabilities and Exposures*. http://cve.mitre.org/ (2011).

[MITRE 2011b]
MITRE. *Common Configuration Enumeration*. http://cce.mitre.org/ (2011).

[Waltermire 2011]
Waltermire, D., Johnson, C., Kerr, M., Wojcik, M., & Wunder, J. *Proposed Open Specifications for Enterprise Information Security Remediation – Draft* (NIST Interagency Report 7670). NIST, 2011.

[Wojcik 2009]
Wojcik, M. N., Wunder, J., Kerr, M., & Waltermire, D. *Proposed Open Specifications for Enterprise Information Security Remediation.* The MITRE Corporation, 2009.

Relevant Websites

Abbreviation	Title	URL
ARF	Assessment Results Format (DoD version 0.41)	http://metadata.dod.mil/mdr/ns/netops/shared_data/arf_index_page/0.41
CCE	Common Configuration Enumeration	http://cce.mitre.org/
CPE	Common Platform Enumeration	http://cpe.mitre.org/
CVE	Common Vulnerabilities and Exposures	http://cve.mitre.org/
NIST CSD	National Institute of Standards and Technology Computer Security Division	http://csrc.nist.gov/
NVD	National Vulnerability Database	http://nvd.nist.gov/
SCAP	Security Content Automation Protocol	http://scap.nist.gov/
XCCDF	eXtensible Configuration Checklist Description Format, a specification language for writing security checklists, benchmarks, and related kinds of documents.	http://scap.nist.gov/specifications/xccdf/

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave Blank)	2. REPORT DATE July 2011	3. REPORT TYPE AND DATES COVERED Final
4. TITLE AND SUBTITLE Standards-Based Automated Remediation: A Remediation Manager Reference Implementation		5. FUNDING NUMBERS FA8721-05-C-0003
6. AUTHOR(S) Sagar Chaki, Rita Creel, Jeff Davenport, Mike Kinney, Benjamin McCormick, Mary Popeck		
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Software Engineering Institute Carnegie Mellon University Pittsburgh, PA 15213		8. PERFORMING ORGANIZATION REPORT NUMBER CMU/SEI-2011-SR-007
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) HQ ESC/XPK 5 Eglin Street Hanscom AFB, MA 01731-2116		10. SPONSORING/MONITORING AGENCY REPORT NUMBER
11. SUPPLEMENTARY NOTES		
12A DISTRIBUTION/AVAILABILITY STATEMENT Unclassified/Unlimited, DTIC, NTIS		12B DISTRIBUTION CODE

13. ABSTRACT (MAXIMUM 200 WORDS)

This report describes the Software Engineering Institute's work in calendar year 2010 for the National Security Agency Computer Network Defense Research and Technology Program Management Office to develop standards for remediation of vulnerabilities and compliance issues on Department of Defense (DoD) networked systems. The overall goals are to assist in the development of remediation standards, demonstrate the functionality DoD would like in a remediation manager, and increase efficiency and effectiveness of remediation by automating the remediation process.

The 2010 Remediation Manager reference implementation demonstrates the following potential applications of remediation and other security automation standards: (1) Ingest scan findings in Security Content Automation Protocol (SCAP) format, extracting host compliance issues (in Common Configuration Enumeration [CCE] format) and vulnerabilities (in Common Vulnerability Enumerations [CVE] format). (2) Map CCE and CVE to remediation actions (in Common Remediation Enumeration [CRE] format). (3) Build remediation tasks in Remediation Tasking Language (RTL), based on CRE. (4) Transmit remediation tasks to a Remediation Tool on a host system. (5) Receive remediation task execution status, in RTL Results Format, from the Remediation Tool. This report identifies capabilities considered for future versions of the reference implementation and the operational system as well as challenges for future work.

14. SUBJECT TERMS automated remediation, computer security, configuration compliance, information assurance, open remediation specification, policy-based remediation, remediation, Remediation Manager, remediation policy, Remediation Tool, Security Content Automation Protocol (SCAP), security noncompliance, security automation standards, standards-based automated remediation, vulnerability	15. NUMBER OF PAGES 80
16. PRICE CODE	

17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT UL